Connecting health care professionals : studies on a generic EHR system framework : how generic can you get?

Connecting health care professionals: studies on a generic EHR system framework

Dit project wordt mogelijk gemaakt door: ZonMw

This dissertation was supported by the School for Public Health and Primary Care (CAPHRI) of the Faculty of Health, Medicne and Life Sciences (FHML) of Maastricht University

Omslagontwerp en boekverzorging Jos Bruystens, Maastricht
Druk Wilco, Amersfoort
ISBN 978-90-9024553-9

Connecting health care professionals: studies on a generic EHR system framework

How generic can you get?

PROEFSCHRIFT
ter verkrijging van de graad van doctor
aan de Universiteit Maastricht,
op gezag van de Rector Magnificus,
Prof. mr. G.P.M.F. Mols,
volgens het besluit van het College van Decanen,
in het openbaar te verdedigen
op vrijdag 16 oktober 2009 om 12.00 uur

door

Anna Wilhelmina Nicole van der Linden

Promotor

Prof. dr. ir. A. Hasman (Universiteit van Amsterdam)

Copromotor

Dr. ir. J.L. Talmon

Beoordelingscommissie

Prof. dr. G.G. van Merode (voorzitter)
Prof. J. Grimson (Trinity College, Dublin, Ireland) PhD
Prof. dr. J. van der Lei (Erasmus Universiteit Rotterdam)
Prof. dr. M. Limburg
Prof. dr. L. de Witte

Contents

Origami

The title pages of the chapters show origami diagrams of the Sonobe element, several variations and constructions that can be built with them. Together they visualize the concept of archetypes that runs like a thread through this dissertation: with a limited set of elements a wide variety of constructions can be built, merely by using different combination methods.

De titelpagina's van de verschillende hoofdstukken tonen origami diagrammen van het Sonobe element, verschillende varianten en diverse constructies die daarmee te maken zijn. Ze vormen samen een verbeelding van het concept van archetypes dat als een rode draad door dit proefschrift loopt: met een beperkte verzameling elementen kunnen uiteenlopende resultaten bereikt worden, door de verschillende manieren van combineren.

Credits go to Dennis Walker for coming up with the modular origami idea. Mrs. Meenakshi Mukerji and Mrs. Tomoko Fuse kindly granted permission for the use of some of their diagrams.

Marvelous Modular Origami
Meenakshi Mukerji
2007, ISB 1-56881-316-3
(Chapters 2, 3, 4, 5, 7)

Floral Origami Globes
Tomoko Fuse
2006, ISBN 978-4-88996-213-0
(Chapters 8, 9, Appendices)

Flat sonobe, credited to Tomoko Fuse
(Chapter 6)

Chapter 1

Introduction

Introduction

This thesis proposes an architectural framework for a generic electronic health record (EHR) system. The architecture should be capable of supporting inter-organizational information exchange while retaining domain independence and resilience to change over time.

1 Definition of the term EHR

From the onset of electronic record keeping several definitions for the electronic equivalent of the paper based medical record emerged. Although in general they were synonyms, they discriminated between the origin of the information (e.g. scanned documents as images versus computer-readable information) and/or the scope of the information (ranging from information generated within one care enterprise to any information relevant to the health of the patient). [1] The most generic definition of the EHR is found in the ASTM E1769 standard: "A comprehensive, structured set of clinical, demographic, environmental, social, and financial data and information in electronic form, documenting the health care given to a single individual." [2] Recent developments in the field have led to an update of the definition of the EHR in the ISO TR 20514 in 2005: "A repository of information regarding the health status of a subject of care in computer processable form, stored and transmitted securely, and accessible by multiple authorised users. It has a standardised or commonly agreed logical information model which is independent of EHR systems. Its primary purpose is the support of continuing, efficient and quality integrated health care and it contains information which is retrospective, concurrent, and prospective." [3]

Literature as well as this thesis use various acronyms to describe the medical information of a patient recorded in some digital form. [4] To add to the confusion, the term EHR is in some context used to refer to the (software) system as well. Despite the differences in terminology, which have been retained in this thesis in order to be faithful to the authors of the referenced publication, they may be interpreted as synonyms for the EHR in the context of this thesis.

The ISO definition clearly demonstrates the trend to use the term EHR to refer to the repository of information. This thesis focuses on a generic system architecture that can support and manage this repository, rather than the structure of the repository itself.

2 Description of the problem addressed

Our current information-aware society with its 24/7 information needs requires ubiquitous access to information. Because of changed behavior in seeking medical help and the greater mobility of patients, physicians have to access health information of their patients residing in various EHR systems, within and outside the physician's organization. Continuous quality improvements of care and research require lifelong patient records to allow learning from the past and learning from mistakes. All these requirements should be met by the EHR systems.

In the beginning of the 1990s the concept of transmural care was introduced in Dutch health care. Transmural care crosses the traditional division between general primary care and specialized hospital care and is comparable to the concepts of 'shared care' or 'integrated care delivery' [5]. Its purpose is to increase the quality of care as well as its efficiency, by promoting and supporting collaboration and coordination between multiple disciplines. [5, 6] Transmural care can be defined as a collaboration between hospital clinicians, GPs and other therapists to provide integrated care to the patient for a particular health problem.
A study by Sas et al [6] showed that transmural care suffers from a lack of overview, due to fragmented information entered in different record systems. If a multi-disciplinary record is available at all, it is often a paper-based record located in a central place, often the patient's home.

In 1996 a project was started to optimize the care of stroke patients from the admission in the hospital to the rehabilitation phase [7]. The vision of this project was to rehabilitate patients in their home environment supported by a multidisciplinary care team of therapists and nurses whenever possible. In this environment a paper-based stroke therapy record was introduced.

In this context, the PropeR project started in 2000. It focused initially on the use of an EHR system and its impact on the quality of care. None of the existing EHR systems in 2000 were suitable for this task. Most were directed towards one single discipline and were hardly able to communicate with other record systems used in the various disciplines involved. This lack of interoperability of systems let us to investigate to what extent a general domain agnostic EHR system can be designed and developed. On the one hand, this system should support clinical care in the settings described above and on the other hand facilitate research on EHR systems in clinical environments without the necessity to redesign the system for each specific research project.

3 Four aspects of an EHR system

Our objective was to develop a domain agnostic EHR system, based on a generic framework. Four aspects for the framework were considered, three directed towards generality and reuse and the fourth focusing in particular on the user interactions with the system:

- **Functional requirements of the EHR.** Analysis of requirements provides insight in the essential functional features that should be implemented in a generic EHR system. These requirements should be as generic and domain agnostic as possible.
- **Architecture.** The design of a generic, reusable architecture starts with a study into the available architectural descriptions. Common architectures provide a grouping of the functional requirements.
- **Standards.** The system should be based on standards, with the main idea that standards help to design systems that can potentially be integrated in a larger environment and which will adhere to "good practice". [8]
- **User interface.** A fourth aspect was added, since the user interface is the only direct interaction of users with the system and will therefore have an important effect on the user acceptance and usefulness of the system. Wyatt et al. argued that 'paper records are organized to assist data entry, not retrieval' [9]. This is a motivation to have the user interface — for both data entry and data retrieval — as a separate aspect to investigate.

These four aspects were considered to elicit a generic set of requirements for the PropeR EHR system.

4 Aim of this thesis

This study focuses on the following research questions.

- Is it possible to define a generic architectural framework for a domain agnostic EHR system?
- Is it possible to implement such an architectural framework?
- Does this framework remain valid over time?

These questions were addressed by performing a literature review to define a set of requirements for a generic EHR system, considering the four aspects described before. Based on these requirements a system — named PropeRWeb — was designed, built and tested to evaluate the possibility of software implementation.

The generic aspect of the framework was studied by widening the scope in various directions: (a) across organizational boundaries and the impact on security, (b) across domain boundaries and the impact on the GUI and finally (c) across time and the impact of the evolution of the medical informatics field on the validity of the framework.

5 Outline of the thesis

The thesis is divided in two parts. The first part, chapters 2 to 5, covers the design, development and evaluation of PropeRWeb. The second part, chapters 6 to 9, discusses the validity of the PropeRWeb architecture from different points of view.

In the first part, the focus lies on the PropeRWeb software. Chapter 2 describes the existing knowledge that was available in the literature at the time of the design of the PropeR EHR framework. A literature review was performed to provide an overview of the major projects and emerging standards. The chapter ends with a list of requirements for the implementation of a new EHR. Given this list of requirements, chapter 3 describes the design considerations and the resulting design of the PropeR EHR.
The implementation of the design is discussed in chapter 4. The first practical results are also presented in this chapter. The software is evaluated on three aspects: usability, domain independency/reusability and appropriateness of the technical approach taken. To evaluate usability, interviews with the test users were held. Domain independency was evaluated by configuration of the software for a few other domains by various persons. The technical evaluation has a reflective nature, discussing the approach taken in the light of the then existing technologies and good practices. The results of these evaluations are presented in chapter 5.

In the second part, the scope is widened. Chapter 6 widens the scope in time. A literature review of the period after the PropeRWeb implementation presents the evolution of the field and discusses the validity of the PropeRWeb architecture based on other evidence emerging from the field over this period. Chapter 7 addresses issues that occur when EHR systems are exchanging information across institutional boundaries. The focus is on security. Chapter 8 widens the scope in user interaction and domain independency. This chapter explores the concept of a generic graphical user interface framework that is part of the overall generic architecture.

Chapter 9 discusses the key lessons learned and provides a description of the architecture using current state of the art techniques. It concludes with suggestions for future work.

6 References

1 Waegemann P. What is an Electronic Patient Record? 1995 [cited 2008/12/15]; Available from: http://web.archive.org/web/19970412211848/www.medrecinst.com/levels.html

2 ASTM E1769-95 Standard Guide for Properties of Electronic Health Records and Record Systems.: ASTM; 1995.

3 Health informatics – Electronic health record – Definition, scope and context: ISO; 2005. Report No.: ISO/TR 20514:2005.

4 Hasman A. Care for records for care. Int J Biomed Comput. 1996 Jul;42(1-2):1-7.

5 van der Linden BA. Transmural Care. In: Schrijvers A, Kodner L, editors. Health and Health Care in the Netherlands. Utrecht: De Tijdstroom; 1997. p. 181-8.

6 Sas P, Tange H, Beusmans G, Crebolder HF, Hasman A. Electronic decision support and reporting: a literature review [In Dutch]. TSG. 2001;79(7):336-41.

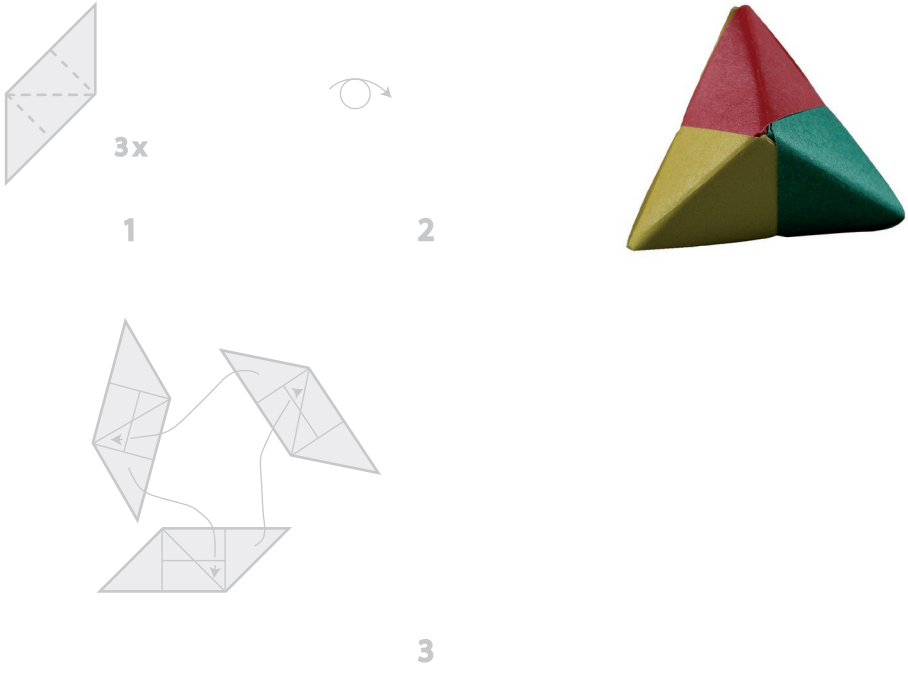
7 Transmural Care Model CVA Region Heuvelland. A model for improvement of the quality of care for CVA patients in the region Heuvelland. [In Dutch]. Maastricht: Synchron; 1996.

8 Neame RL. The crucial roles of standards and strategy in developing a regional health information network. Int J Biomed Comput. 1995 Oct 1;40(2):95-100.

9 Wyatt JC, Wright P. Design should help use of patients' data. Lancet. 1998 Oct 24;352(9137):1375-8.

Chapter 2

Requirements for a generic system



Requirements for a generic system

1 Introduction

The main goal of the PropeR project was to develop a generic, domain independent electronic health record (EHR) system. It should be easy to tailor the system to a specific domain and to adapt it to future research needs.
Several authors such as Neame and Kohane et al. [1, 2] have pointed out the advantages of the use of standards, such as cost-effective and faster development. Other areas of technology show already time-tested evidence of the benefits of standards, ranging from screw sizes to the way a car is operated to units of measurement. This lead to the decision to pursue a standards compliant system.
At the start of the project, the majority of the electronic information systems implemented in hospitals, were focused on supporting administrative transactions (such as billing), rather than direct support of the care process of the patient. More clinically oriented systems in use were special purpose systems such as Laboratory Information Systems (LIS), Picture Archiving and Communication Systems (PACS) and, outside the hospital, general GP systems. None of these systems provided the kind of information (i.e. primary care, but non-GP) or the functionality (i.e. multidisciplinary) for the needs of the project.
This chapter analyzes in more detail what was known in the literature on EHR system architecture at the beginning of the PropeR project to define the requirements of the system to be developed. The literature review in this chapter covers the period from 1995-2001. The timeframe encompasses five years before the start of the PropeR project and includes the first two years of the project. The year 1994 was subsequently added, to include a few key articles such as Frandji et al. on the RICHE project [3].

2 Methods

To avoid missing key articles in the domain all references indexed in PUBMED with the MeSH term 'Medical Record Systems, Computerized' limited by the date range 1994 – 2001, with an abstract and written in English were retrieved. This resulted in 2923 articles.

As stated in chapter 1, four aspects are of particular importance for the design of a generic system: Architecture, Requirements, Standards and User Interface.

Common architectures provide a reference framework for an EHR system. Analysis of requirements provides insight in the essential functional features that should be implemented in a generic EHR. Standards help to develop systems that can potentially be integrated in a larger environment and which will adhere to "good practice". The user interface is the only direct interaction of users with the system and will therefore have an effect on the user acceptance and usefulness of the system.

In a first phase, the 2923 papers were reduced by assessing their content mainly based on their titles. Papers were also categorized in the four aspects. Papers in the Architecture category should have the description of the architecture of the EHR system as the main or one of the major topics. The described system should be an EHR system, primarily used for clinical purposes; possible usage for research is of secondary interest. Articles focusing on referencing medical information sources (such as PubMed) as main usage are excluded. The Requirements category holds papers that describe requirements for a successful EHR system. The Standards category includes papers that describe a standard for EHR systems, such as HL7 or CEN TC 251 13606, or a 'de facto' standard. Papers describing standardization efforts on certain aspects such as security are also included. The User Interface category holds papers that describe general aspects of user interfaces with a particular interest in approaches for generic and flexible interfaces.
This first analysis reduced the base set to 353 papers. Next, the titles and abstracts of those papers were studied in detail by two independent scorers. The objective was to find papers to support the goal of this review: an assessment of the state of the art in the period 1994 – 2001 on the four aspects selected. Discrepancies were discussed and resolved. This resulted in a set of 146 potentially interesting papers. After reading the full paper, this set was further reduced to 75 articles that contained relevant information for the purpose of this chapter. Based on these articles 6 additional articles were retrieved and added to the set.

The Papers [4] software for the management, evaluation and annotation of the papers was used.

3 Results

Table 1 shows the breakdown of the final 81 articles over the four categories.

TABLE 1 Division of articles in categories

Category	Initial screening	After consensus	Articles with relevant information
Standards	34	15	10
Requirements	35	12	5
Architecture	229	85	45
User Interface	55	34	21
Total	353	146	81

3.1 STANDARDS AND ARCHITECTURES

Neame defines a standard as "that which is established by competent authority, public opinion, custom or general consent as a rule, model or measure capable of satisfying certain requirements" [2]. Various authors have stressed the importance of adhering to standards [1, 2] and even suggested the definition of standards for areas of the field where standards were lacking. [5]

This chapter uses the term *standards* for two kinds of specifications: a 'true' standard as defined by a standards development organization (SDO), even if the standard is still in draft mode, and a 'de facto' standard, usually a methodology that is adopted as 'good practice' by others.

The Standards category consists of ten articles, which discuss the standards in it self. [2, 6-14] The Architecture category contains several articles that discuss an implementation of one of these standards, which is why they are discussed together in this section.

The standards under development in the timeframe studied (1994 – 2001) can be divided in three subsets:

- System architecture
- Clinical data representation
- Clinical messaging

3.1.1 System architecture

Standards on system architecture focus on integrating various medical systems, usually legacy systems, to provide the virtual patient record, as described in the Requirements section (§ 3.2.2).

The major European standard in this context is the Healthcare Information Systems Architecture (HISA) registered as CEN ENV 12967 [9]. This standard was assembled based on the findings of various successive research projects. The chain of projects started with Réseau d'Information et de Communication Hospitalier Européen (RICHE). RICHE defined a reference architecture, an open framework for integrated health care systems. [3] The architecture combines distributed components in sev-

eral layers to provide an integrated view of the available information on a patient. The RICHE architecture was implemented in the NUCLEUS and EDITH projects [9, 15]. Subsequent projects such as HANSA resulted in the commercial Distributed Healthcare Environment (DHE) [9]. Results and experiences from these projects informed and revised the HISA standard. DHE is considered a reference implementation of HISA.

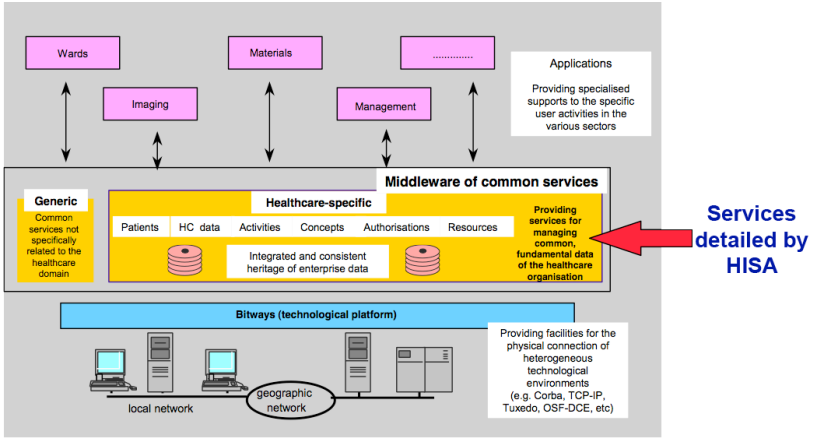
The Object Management Group (OMG) followed a different approach. The OMG is a non-profit computer industry consortium, developing a wide range of enterprise integration standards. The most well known is the Common Object Request Broker Architecture (CORBA) [16]. The CorbaMed domain taskforce (later renamed to Healthcare Domain Taskforce, HDTF) published specifications intended to solve generic problems in the healthcare domain, such as the Person Identification Service (PIDS) [17] and the Clinical Observation Access Service (COAS) [18].

HISA focused on an open system architecture of a middleware layer (see Figure 1). This layer combines the data from several legacy systems using different technological environments to provide an integrated view of the available patient information, which can be accessed by specialized applications. The middleware layer provides high level, abstract services, called Health care Common Services (HCS), such as Subject of Care HCS (managing non-clinical information on patients).
CorbaMed on the other hand, starts with CORBA as the middleware layer and provides generic services that comply with the interoperability features provided by CORBA. These specifications are more implementation oriented than the more specification oriented HISA. The CORBA services are defined in a platform-independent language (ICL). Software in various programming languages exists which can convert ICL into the respective programming language. HISA on the other hand specifies at a higher level the functionality of the various components.
Following the OMG's process standards the CorbaMed specifications are available in source code. This not only allows for easy testing, but also provides insight in the meaning and the feasibility of the specifications. In contrast, the implementations of HISA are not open (i.e. available to the public for inspection and modification). In fact, the only available implementation is the commercial DHE.

3.1.2 Clinical data representation

Standardization work on clinical data representation focused on structuring generic EHR concepts and on knowledge representation.
The Good European Health Record (GEHR) project, which was funded from 1992 to 1994, defined an architecture for comprehensive, communicative and portable

FIGURE 1 HISA architecture [9, 19]



electronic health records. From this work, the term "health record item" (HRI) was defined as the smallest unit of information, which remains meaningful as an entry in a healthcare record [20, 21]. The GEHR project was followed by Synapses, which implemented the GEHR architecture in a federated server, which supported the Synapses Federated Healthcare Record, which in turn was based on the Synapses Object Model (SynOM) and the Synapses Object Dictionary (SynOD). [22, 23]
A follow-up to the Synapses project was the Synex project [24], which integrated the Synapses federated server with DHE to provide an integration platform for new and legacy applications.

The work of Moorman et al. [25] on descriptive knowledge informed the data dictionary model (SynOD) in Synapses[1].
In the timeframe of the Synapses and Synex projects, the Good electronic Health Record (GeHR) was funded in Australia. It extended the original GEHR project and developed the concept of archetypes, defined as constraint descriptions and rules for valid clinical structures, which can be built with concrete classes [26].
These archetypes and the two-model approach underpinning them resembled the SynOM/SynOD separation of the Synapses project. In retrospect, the concepts developed by Moorman et al. could be considered as 'simple archetypes'.

In 2001, the *open*EHR foundation was founded to work on consolidating and implementing the knowledge gained in the GEHR, Synapses/Synex and GeHR projects. They also further developed the archetype concept.

1 D. Kalra in personal conversation.

In the same timeframe, a special interest group from HL7 developed what has later become the Clinical Document Architecture (CDA). CDA can be considered as a mix of clinical data representation and clinical messaging. The data structure builds on classes from the HL7 Reference Information Model (RIM), while the purpose of CDA was the information exchange in the form of clinical documents.
Various standards emerged in the same time frame as these projects. The Electronic Healthcare Record Architecture (EHCR-A, registered as CEN-ENV 12265) influenced the Electronic Health Record Communication (EHRCom, CEN-ENV 13606) [13] standard and was later replaced by it. This was caused by the cross-pollination of ideas between the research projects and the work on the standards.
The ISO TS 18308, published in 2002, provided an overview of requirements for an Electronic Health Record Reference Architecture, compiled from numerous descriptions of projects such as the GEHR and Synapses project. Note that this technical specification focuses on the *record* architecture, not on the *system* architecture.

3.1.3 Clinical messaging

Clinical messaging considers EHR systems as black boxes and focuses on standardizing the information exchange between the systems. The most prominent standards in this context were HL7 Message Framework, UN/EDIFACT, GPICs, MML and specialized standards such as DICOM.
The HL7 Message Framework version 2 was an American standard for information exchange between clinical systems; UN/EDIFACT its European counterpart.
The HL7v2 framework was widely adopted at the time, but presented some important drawbacks, such as lack of rigorousness to avoid ambiguity in the implementation, resulting in vendor-dependent implementations. EDIFACT presented similar drawbacks. The problems of HL7v2 resulted in the development of HL7 version 3, starting in the late 1990s. HL7v3 was a major overhaul of the work done in version 2, starting from a different point of view: a model centric approach which resulted in a formalized Reference Information Model (RIM) and a formalized modeling process, the HL7 Message Development Framework (HL7-MDF) [7].
The HL7-MDF describes the development process to define standardized messages based on a higher-level domain model. During the message definition process common structures were recognized and formalized for reuse, referred to as Common Message Element Types or CMETs.
The objective of the General Purpose Information Component (GPIC) standard, developed in Europe as CEN prEN 14822, was to define common, reusable information components. GPICs are similar to the HL7 CMETs. In 2000 harmonization efforts of GPICs and CMETs were started.
In Japan the Medical Markup Language (MML) was developed. MML is based on

XML and focuses on structuring the transfer of comprehensive medical information on patient referral [6]. MML is similar to HL7v3, but HL7v3 focuses on formalizing specific messages that comply to use cases, while MML strives to minimize the number of different messages.
Specialized standards such as DICOM, which focus on a single aspect of information communication, images in the case of DICOM, are outside the scope of this chapter.

A more elaborate discussion of the history and relationships of the various projects can be found in chapter 5 of the thesis by Kalra [27].

3.2 REQUIREMENTS FOR AN EHR SYSTEM

The Requirements category contained five articles. These articles [5, 15, 28-30] all describe requirements to be fulfilled by successful EHR systems. These requirements can be categorized as:

- Patient centered
- Integration of multiple information sources
- Value added information processing
- Secure
- Flexible, easy to use user interface
- Multiprofessional
- Secondary use of data

3.2.1 Patient centered

The focus of the system should be on providing information supporting care and wellness of the patient, rather than on the support of organizational administrative transactions such as billing and scheduling. [28, 29]

3.2.2 Integration of multiple information sources

The system should present the information to the user as coming from one source, while the underlying information is retrieved from several sources within the organization or even from outside the organization. [28, 29] This requires standards for information exchange as well as controlled access to sensitive information sources. [29]
Another source of information is general medical literature. Integration of sources like PubMed can improve medical treatment. [29]
The integration of multiple information sources is often referred to as the virtual patient record. This term will be used in the remainder of the thesis to refer to the

concept of an EHR record that integrates patient record information of various information sources but is presented to the user as a single patient record.

3.2.3 Value added information processing

To support the clinician in making better decisions, additional functionality needs to be added. This functionality can consist of decision support in various forms such as reminders of adverse drug events, dosage calculations or duplicate test reminders. [29]

3.2.4 Security

Sensitive information such as a patient's health information should be protected from unwanted access. [5, 15, 29, 30] Unwanted access in this context is defined as all persons within or outside the organization who are not involved in the treatment of the patient. It therefore not only includes non-medical persons but also medical professionals without a specific care relationship with the patient.

A secure system ensures controlled access, prevents user errors and provides a trustworthy system.

Controlled access consists of:

- Identification,
- Authentication,
- Authorization,
- Data integrity (including prevention of user errors),
- Non-repudiation.

3.2.5 Flexible, easy to use user interface

The majority of the users interact with the EHR system only through its user interface (UI). By making the UI easy to use and easily adaptable (by the end-user or by the developers) to changing work processes and personal preferences, the overall acceptance of the system will increase. [28, 30]

3.2.6 Multiprofessional

The system should be able to accommodate not only simultaneous access from multiple users at the same or different locations [30], but also be able to store and retrieve information from different medical disciplines. By integrating information of different disciplines, it is not only possible to give the clinician a better overview of the patient's information, but also help in reduction of ordering redundant tests.

3.2.7 Secondary use of the information

The information in EHR systems is primarily used for patient care and medical treatment. However, the acceptance and usefulness of the systems and the information is increased when secondary uses can be supported as well. Key uses are administrative uses such as billing and policy decisions, and medical research. [29]
These uses require standardization of the vocabulary and codes used. [29, 30]

At the time, a large body of literature on requirements of EHRs could be found in project reports. Although they are not part of the papers discussed here, they are included for the information on requirements. The most prominent projects were GEHR and Synapses, which were discussed more elaborately in § 3.1.2 and the EHCR Support Action project (EHCR-SupA) [31]. The latter provided a summary of earlier projects such as GEHR in a consolidated list of requirements, most of them for the record architecture and only a few for the systems, all of which are addressed by the list above.
As stated in § 3.1.2 these projects fed into the ISO TS 18308 standard that provides an overview of the requirements for the record architecture. [32]

An important aspect of the various requirements described is the fact that they often can only be fulfilled if appropriate standards are available.

3.3 SYSTEM ARCHITECTURES

The 45 articles in the Architecture category were selected for their description of the underlying architecture of an EHR system. [1, 3, 20-24, 33-70] The majority of the architectures are based on a client-server architecture, usually with a middleware layer. From 1997 on, web based systems become more prominent.
Spahni et al. [69] give an overview of the importance of a middleware layer. They define a middleware service as "a general purpose service that sits between platforms and applications". The middleware layer is essential in a distributed environment to enable cost-effective, user-oriented applications that are based on an integrated view of the underlying heterogeneous databases.
Several architectures such as the RICHE architecture and the Synapses architecture, are structured following the guidelines of the Reference Model of Open Distributed Processing (RM-ODP) [71]. This model differentiates various viewpoints on the system to build and provide a separation in achievable parts that allows more complete descriptions of the functionality of the system. Therefore, this model provides a good starting point for the design of a generic EHR system.
The HELIOS project, which started in 1991, focused on the development of an

object-oriented software engineering environment (SEE). [38] "The HELIOS SEE is constituted from several independent and cooperative pieces of software, called software components, devoted to the management of particular modalities of medical information both when applications are being developed and being run." The kernel consists of mandatory components that focus on development, data integration and presentation integration. The services consist of domain-oriented components, such as a component to provide a mechanism for data-driven decision support. The kernel and the services are connected through the HELIOS Unification Bus, a middleware layer, similar to CORBA, which provides a connection of the components and a message routing mechanism. The ARTEMIS-2 project is an implementation example of the HELIOS architecture. [66]

Nine articles discuss several projects based on the HISA standard. As stated in the previous section, the RICHE project [3, 70] as well as the NUCLEUS project and the DHE are all based on and have influenced the HISA standard. The Synapses project [22, 23, 60] and its successor the Synex project [24] are also based on the HISA standard. The focus of the Synapses and Synex projects is on the federated approach. Several systems are connected to provide a virtual system. Queries to the virtual system are parsed and propagated to the connected systems. A 'store and forward' layer holds the results of these queries to allow quick retrieval of frequently requested information. A modern day analogy is the proxy server that caches frequently used web pages.

The HISA standard defines a middleware layer that consists of a 'bitways' layer, the technical infrastructure to connect the various distributed (legacy) systems, and a services layer that integrates the information from the systems.

A different approach is the CorbaMed components, based on CORBA and implemented in several projects such as the TeleMed project [37, 39], and its successor the OpenEMed project [72]. The TeleMed project is an implementation of the CorbaMed services to provide simultaneous access and editing facilities to a virtual patient record by several physicians in remote locations. The sources of this patient record are connected through a CORBA layer, while the GUI is web based. HYGEIAnet, a Greek project, defines a reference architecture for a regional health care network based on CORBA and CorbaMed components [59, 63, 73]. XML is used as the COAS information structure to transport the information from underlying databases to the client application.

An important Dutch project is the Open Record of CAre (ORCA) project [54]. The architecture is much more focused on structured data entry than on the integration of distributed systems as the other projects are. The interesting aspect of this architecture is the fact that it separates a knowledge graph from the supporting system which could be considered as a very early two-model approach.

The emerging web based applications are best represented by the W3-EMRS project [65] and the CareWeb project, an implementation of W3-EMRS. [40]. The W3-EMRS architecture is also focused on integration of distributed systems, but with a strong use of web technologies. The Agglutinator collects information from different sites, which are accessed through web services, based on HL7 queries. The collected data is then formatted and converted into HTML and presented in a user's browser. The CareWeb implementation adds security and auditing to the system.
Another web-based project is the Personal Internetworked Notary and Guardian (PING) project [43]. This project defines a secure, distributed, user controlled data-storage system. In this context, the user is the patient. The PING architecture integrates information from various sources by importing the information upon the patient's request.

3.4 USER INTERFACE

With the introduction of Windows 95, graphical user interfaces (GUI) became mainstream and a prerequisite of modern systems. At the same time, the Internet became accessible for a large number of people. Advantages of web-based applications in general and for health-related purposes in particular were quickly identified, as Cimino demonstrated by the rapidly increasing number of scientific articles with internet-related topics [74].

The user interface category has 21 papers [25, 75-94], which can be subdivided in web based GUIs and special GUIs. The first subset describes various approaches to create a GUI using web pages. This GUI often follows a common approach by presenting a list of key-value pairs. Sometimes the multimedia capabilities of the platform are used for display of images and graphs.
The second subset describes GUIs with a more uncommon display of information. These GUIs are usually implemented in stand-alone applications. Examples in the second subset are timeline oriented GUIs [75, 80, 87, 89] and adaptive GUIs, where domain knowledge and user actions are used to calculate an optimal order of menu items [91, 92] or predictions for data entry [90].

The web platform shows some drawbacks when compared to a desktop application. The most prominent are the limits of HTML and the statelessness of the HTTP protocol. However, the advantages, such as flexible and easy development and ubiquitous use, outweigh these limitations.

4 List of requirements

Literature shows that a modern EHR system is based on standards, on components and is web based. Client-server architecture is still supported, but the client is now a web browser.

This leads to the following list of requirements for our system:

- The software should be flexible enough to accommodate changing user requirements without a large programming effort (§3.3).
- The User Interface should be graphical and flexible (§3.2.5, §3.4).
- The EHR system should be robust, i.e. designed, implemented and tested according to state of the art software engineering methodologies.
- Current security standards should be implemented (§3.2.4).
- Current privacy regulations should be adhered to (§3.2.4).
- The software should be based on relevant national and/or international standards (§3.1.1, §3.2).
- The software should be generic enough to be usable in other domains and by other projects (§3.3). This is extended to include the following requirements:
  - The software should be open source.
  - The software should be platform-independent.
- Data/software should be accessible from different locations (§3.2.6).
- Data from different sources and disciplines should be handled (§3.2.2, §3.2.6).

The next chapter will focus on the design of the PropeR system. The design considerations are based on the above requirements as determined from the literature.

5 References

1 Kohane IS, Greenspun P, Fackler JC, Cimino C, Szolovits P. Building national electronic medical record systems via the World Wide Web. J Am Med Inform Assoc. 1996 May 1;3(3):191-207.

2 Neame RL. The crucial roles of standards and strategy in developing a regional health information network. Int J Biomed Comput. 1995 Oct 1;40(2):95-100.

3 Frandji B, Schot J, Joubert M, Soady I, Kilsdonk A. The RICHE Reference Architecture. Medical informatics = Médecine et informatique. 1994 Jan 1;19(1):1-11.

4 Papers. [cited 2009/01/15]; 1.8.6: Available from: http://mekentosj.com/papers/

5 de Meyer F, Lundgren PA, de Moor GJ, Fiers T. Determination of user requirements for the secure communication of electronic medical record information. Int J Med Inform. 1998 Mar 1;49(1):125-30.

6 Araki K, Ohashi K, Yamazaki S, Hirose Y, Yamashita Y, Yamamoto R, et al. Medical markup language (MML) for XML-based hospital information interchange. J Med Syst. 2000 Jun 1;24(3):195-211.

7 Beeler GW. Taking HL7 to the next level. MD Comput. 1999 Mar 1;16(2):21-4.

8 Dolin RH, Alschuler L, Beebe C, Biron PV, Boyer SL, Essin DJ, et al. The HL7 Clinical Document Architecture. J Am Med Inform Assoc. 2001 Nov 1;8(6):552-69.

9 Ferrara FM. The standard 'Healthcare Information Systems Architecture' and the DHE middleware. Int J Med Inform. 1998 Oct 1;52(1-3):39-51.
10 Ferrara FM, Sottile PA, Grimson W. The holistic architectural approach to integrating the healthcare record in the overall information system. Stud Health Technol Inform. 1999;68:847-52.
11 Liu GC, Cooper JG, Schoeffler KM, Hammond WE. Standards for the electronic health record, emerging from health care's Tower of Babel. Proc AMIA Symp. 2001:388-92.
12 Louwerse K, van Ditmarsch M, Flikkenschild E. Experiences with a new security standard for healthcare information systems. Stud Health Technol Inform. 1999;68:311-4.
13 Markwell D, Fogarty L, Hinchley A. Validation of a European message standard for electronic health records. Stud Health Technol Inform. 1999;68:818-23.
14 Sokolowski RA, Dudeck J. XML and its impact on content and structure in electronic health care documents. Proc AMIA Symp. 1999:147-51.
15 Kilsdonk AC, Frandji B, van der Werff A. The NUCLEUS integrated electronic patient dossier breakthrough and concepts of an open solution. Int J Biomed Comput. 1996 Jul 1;42(1-2):79-89.
16 Object Management Group (OMG). [cited 2008/12/03]; Available from: http://www.omg.org
17 Person Identification Service, version 1.1. [cited 2008/12/03]; Available from: http://www.omg.org/technology/documents/formal/person_identification_service.htm
18 Clinical Observations Access Service, version 1.0. [cited 2008/12/03]; Available from: http://www.omg.org/technology/documents/formal/clinical_observation_access_service.htm
19 Ferrara FM. CEN TC251 WG Meeting Berlin 1997. 1997 [cited 2008/12/03]; Available from: http://www.tc251wgiv.nhs.uk/pages/pdf/141ehisa.pdf
20 Griffith SM, Kalra D, Lloyd DS, Ingram D. A portable communicative architecture for electronic healthcare records: the Good European Healthcare Record project (Aim project A2014). Medinfo. 1995;8 Pt 1:223-6.
21 Kalra D. The Good European Health Record. Computer methods and programs in biomedicine. 1994 Oct 1;45(1-2):83-9.
22 Grimson J, Grimson W, Berry D, Stephens G, Felton E, Kalra D, et al. A CORBA-based integration of distributed electronic healthcare records using the synapses approach. IEEE Trans Inf Technol Biomed. 1998 Sep 1;2(3):124-38.
23 Grimson W, Berry D, Grimson J, Stephens G, Felton E, Given P, et al. Federated healthcare record server–the Synapses paradigm. Int J Med Inform. 1998 Oct 1;52(1-3):3-27.
24 Xu Y, Sauquet D, Zapletal E, Lemaitre D, Degoulet P. Integration of medical applications: the 'mediator service' of the SynEx platform. Int J Med Inform. 2000 Sep 1;58-59:157-66.
25 Moorman PW, van Ginneken AM, van der Lei J, van Bemmel JH. A model for structured data entry based on explicit descriptional knowledge. Methods Inf Med. 1994 Dec 1;33(5):454-63.
26 Beale T. The GEHR Object Model Architecture, Rev 4.1 draft E. 2000 [cited 2008/12/03]; Available from: http://www.openehr.org/shared-resources/gehr_all/gehr_australia.html
27 Kalra D. Clinical Foundations and Information Architecture for the Implementation of a Federated Health Record Service: University College London; 2002.
28 Degoulet P, Jean FC, Safran C. The health care professional multimedia workstation: development and integration issues. Int J Biomed Comput. 1995 Apr 1;39(1):119-25.
29 Heller EE. The computer-based patient record vision contrasted with HIS/MIS. Int J Biomed Comput. 1995 Apr 1;39(1):19-23.
30 Lundgren PA, Wisser C. Functional requirements for IT support for nursing information systems integrated in electronic healthcare record systems (EHCRS). Stud Health Technol Inform. 1997;46:337-42.
31 Dixon R, Grubb PA, Lloyd D, Kalra D. Consolidated List of Requirements. EHCR Support Action Deliverable 1.4. Brussels: European Commission DGXIII; 2001.
32 ANSI. ISO/TS 18308 Health Informatics - Requirements for an Electronic Health Record Architecture: ISO; 2003.
33 Huff SM, Haug PJ, Stevens LE, Dupont RC, Pryor TA. HELP the next generation: a new client-server architecture. Proc Annu Symp Comput Appl Med Care. 1994:271-5.
34 Findling A, Horsch A, Zink A. An open distributed medical image and signal data server network with World Wide Web front-end. Stud Health Technol Inform. 1997;43 Pt B:591-5.

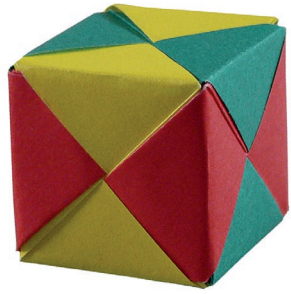
35 Wang DJ, Harkness KB, Allshouse C, Elliot L, Szekalski S, Mandell SF. Development of a web based electronic patient record extending accessibility to clinical information and integrating ancillary applications. Proc AMIA Symp. 1998:131-4.
36 Takeda H, Matsumura Y, Kuwata S, Nakano H, Sakamoto M, Yamamoto R. Architecture for networked electronic patient record systems. Int J Med Inform. 2000 Nov 1;60(2):161-7.
37 Forslund DW, Cook JL. The importance of Java and CORBA in medicine. Proc AMIA Annu Fall Symp. 1997:364-8.
38 Degoulet P, Jean FC, Engelmann U, Meinzer HP, Baud R, Sandblad B, et al. The component-based architecture of the HELIOS medical software engineering environment. Computer methods and programs in biomedicine. 1994 Dec 1;45 Suppl:S1-11.
39 Forslund DW, Phillips RL, Kilman DG, Cook JL. Experiences with a distributed virtual patient record system. Proc AMIA Annu Fall Symp. 1996:483-7.
40 Halamka JD, Osterland C, Safran C. CareWeb, a web-based medical record for an integrated health care delivery system. Int J Med Inform. 1999 Apr 1;54(1):1-8.
41 Cimino JJ, Sengupta S, Clayton PD, Patel VL, Kushniruk AW, Huang XL. Architecture for a Web-based clinical information system that keeps the design open and the access closed. Proc AMIA Symp. 1998:121-5.
42 Kuma H, Tsuchiya Y. Database access method for autonomous distributed total hospital information system and its object-oriented design. Medinfo. 1995;8 Pt 1:387-90.
43 Riva A, Mandl KD, Oh DH, Nigrin DJ, Butte A, Szolovits P, et al. The personal internetworked notary and guardian. Int J Med Inform. 2001 Jun 1;62(1):27-40.
44 Quade G, Novotny J, Burde B, May F, Beck LE, Goldschmidt AJ. Worldwide telemedicine services based on distributed multimedia electronic patient records by using the second generation Web server hyperwave. Proc AMIA Symp. 1999:916-20.
45 Patil R, Silva J, Swartout W. An architecture for a health care provider's workstation. Int J Biomed Comput. 1994 Jan 1;34(1-4):285-99.
46 Baud RH, Rassinoux AM, Lovis C. Document versus data centred approach to the EPR. Stud Health Technol Inform. 1999;68:853-7.
47 Chueh HC, Raila WF, Pappas JJ, Ford M, Zatsman P, Tu J, et al. A component-based, distributed object services architecture for a clinical workstation. Proc AMIA Annu Fall Symp. 1996:638-42.
48 Buffone GJ, Petermann CA, Bobroff RB, Moore DM, Dargahi R, Moreau DR, et al. A proposed architecture for ambulatory systems development. Medinfo. 1995;8 Pt 1:363-6.
49 van Mulligen EM, Timmers T, Brand J, Cornet R, van den Heuvel F, Kalshoven M, et al. HERMES: a health care workstation integration architecture. Int J Biomed Comput. 1994 Jan 1;34(1-4):267-75.
50 Van De Velde R. Framework for a clinical information system. Int J Med Inform. 2000 Jan 1;57(1):57-72.
51 Verderio V, Cooper PA. A core middleware service in H.I.S.: the experience of IC_PIDRM. Stud Health Technol Inform. 2000;77:964-8.
52 Pollard DL, Hammond WE. Object technology: raising the standards for healthcare information systems. Medinfo. 1998;9 Pt 1:217-21.
53 Dargahi R, Fowler J, Moreau DR, Buffone GJ. A server architecture for ambulatory patient record systems. Medinfo. 1995;8 Pt 1:219-22.
54 van Ginneken AM, Stam H, van Mulligen EM, de Wilde M, van Mastrigt R, van Bemmel JH. ORCA: the versatile CPR. Methods Inf Med. 1999 Dec 1;38(4-5):332-8.
55 Chu S, Cesnik B. A three-tier clinical information systems design model. Int J Med Inform. 2000 Jul 1;57(2-3):91-107.
56 Papadakis I, Chrissikopoulos V, Polemi D. A secure Web-based medical digital library architecture based on TTPs. Stud Health Technol Inform. 2000;77:610-6.
57 Klimczak JC, Witten DM, Ruiz M, Mitchell JA, Brilhart JG, Frankenberger ML. Providing location-independent access to patient clinical narratives using Web browsers and a tiered server approach. Proc AMIA Annu Fall Symp. 1996:623-7.
58 Masseroli M, Pinciroli F. Web architecture for the remote browsing and analysis of distributed medical images and data. Medinfo. 2001;10(Pt 1):43-7.
59 Katehakis DG, Sfakianakis S, Tsiknakis M, Orphanoudakis SC. An infrastructure for Integrated

Electronic Health Record services: the role of XML (Extensible Markup Language). J Med Internet Res. 2001 Jan 1;3(1):E7.
60 Grimson W, Sottile PA. Synapses in the context of healthcare information systems. Stud Health Technol Inform. 1997;45:30-9.
61 Dore L, Lavril M, Jean FC, Degoulet P. An object oriented computer-based patient record reference model. Proc Annu Symp Comput Appl Med Care. 1995:377-81.
62 Veloso M, Estevao N, Ferreira P, Rodrigues R, Costa CT, Barahona P. From hospital information system components to the medical record and clinical guidelines & protocols. Stud Health Technol Inform. 1997;43 Pt A:300-4.
63 Katehakis DG, Kostomanolakis S, Tsiknakis M, Orphanoudakis SC. An open, component-based information infrastructure to support integrated regional healthcare networks. Medinfo. 2001;10(Pt 1):18-22.
64 Willard KE, Sielaff BH, Connelly DP. Integrating legacy laboratory information systems into a client-server world: the University of Minnesota Clinical Workstation (CWS) project. Methods Inf Med. 1995 Jun 1;34(3):289-96.
65 van Wingerde FJ, Schindler J, Kilbridge P, Szolovits P, Safran C, Rind DM, et al. Using HL7 and the World Wide Web for unifying patient data from remote databases. Proc AMIA Annu Fall Symp. 1996:643-7.
66 Lemaitre D, Jaulent MC, Gunnel U, Demiris AM, Michel PA, Rassinoux AM, et al. ARTEMIS-2: an application development experiment with the HELIOS environment. Comput Methods Programs Biomed. 1994 Dec 1;45 Suppl:S127-38.
67 Li YC, Kuo HS, Jian WS, Tang DD, Liu CT, Liu LL, et al. Building a generic architecture for medical information exchange among healthcare providers. Int J Med Inform. 2001 May 1;61(2-3):241-6.
68 Grammatikou M, Stamatelopoulos F, Maglaris B. Distributed information system architecture for Primary Health Care. Stud Health Technol Inform. 2000;77:978-82.
69 Spahni S, Scherrer JR, Sauquet D, Sottile PA. Middleware for healthcare information systems. Stud Health Technol Inform. 1998;52 Pt 1:212-6.
70 Frandji B. Open architecture for health care systems: the European RICHE experience. Stud Health Technol Inform. 1997;45:11-23.
71 Putman J, Boehm B. Architecting with RM-ODP: Prentice Hall PTR; 2001.
72 OpenEMed. [cited 2008/12/03]; Available from: http://www.openemed.org/
73 Tsiknakis M, Katehakis DG, Orphanoudakis SC. An open, component-based information infrastructure for integrated health information networks. Int J Med Inform. 2002 Dec 18;68(1-3):3-26.
74 Cimino JJ. Beyond the superhighway: exploiting the Internet with medical informatics. Journal of the American Medical Informatics Association : JAMIA. 1997 Jan 1;4(4):279-84.
75 Plaisant C, Mushlin R, Snyder A, Li J, Heller D, Shneiderman B. LifeLines: using visualization to enhance navigation and analysis of patient records. Proc AMIA Symp. 1998:76-80.
76 van Ginneken AM, Verkoijen MJ. A multi-disciplinary approach to a user interface for structured data entry. Medinfo. 2001;10(Pt 1):693-7.
77 Matsumura Y, Takeda H, Okada T, Kuwata S, Nakazawa H, Hazumi N, et al. Devices for structured data entry in electronic patient record. Medinfo. 1998;9 Pt 1:85-8.
78 Hinds A, Greenspun P, Kohane IS. WHAM!: a forms constructor for medical record access via the World Wide Web. Proc Annu Symp Comput Appl Med Care. 1995:116-20.
79 Kirby J, Rector AL. The PEN&PAD data entry system: from prototype to practical system. Proc AMIA Annu Fall Symp. 1996:709-13.
80 Combi C, Pinciroli F, Musazzi G, Ponti C. Managing and displaying different time granularities of clinical information. Proc Annu Symp Comput Appl Med Care. 1994:954-8.
81 Aisaka K, Tsutsui K, Murakami Y, Ban H, Hashizume A, Oka Y, et al. User interface design and evaluation for electronic medical record system. Medinfo. 1995;8 Pt 1:781-4.
82 Berkowicz DA, Barnett GO, Chueh HC. Component architecture for web based EMR applications. Proc AMIA Symp. 1998:116-20.
83 Simkus R. Application of an intelligent graphical interface to electronic patient records. Medinfo. 2001;10(Pt 1):690-2.
84 Franklin MJ, Sittig DF, Schmiz JL, Spurr CD, Thomas DR, O'Connell EM, et al. Modifiable templates

facilitate customization of physician order entry. Proc AMIA Symp. 1998:315-9.
85 Yoder JW, Schultz DF, Williams BT. The MEDIGATE graphical user interface for entry of physical findings: design principles and implementation. Medical Examination Direct Iconic and Graphic Augmented Text Entry System. J Med Syst. 1998 Oct 1;22(5):325-37.
86 Kolodner RM. Functional workstation requirements: clinical perspectives. Int J Biomed Comput. 1994 Jan 1;34(1-4):115-21.
87 Bui AA, Aberle DR, McNitt-Gray MF, Cardenas AF, Goldin JG. The evolution of an integrated timeline for oncology patient healthcare. Proc AMIA Symp. 1998:165-9.
88 Brown SH, Hardenbrook S, Herrick L, St Onge J, Bailey K, Elkin PL. Usability evaluation of the progress note construction set. Proc AMIA Symp. 2001:76-80.
89 Fackler JC, Kohane IS. Monitor-driven data visualization: SmartDisplay. Proc Annu Symp Comput Appl Med Care. 1994:939-43.
90 Prokosch HU, Amiri F, Krause D, Neek G, Dudeck J. A semantic network model for the medical record of a rheumatology clinic. Medinfo. 1995;8 Pt 1:240-4.
91 DeFriece RJ. Design considerations for intelligent data entry: development of MedIO. Proc Annu Symp Comput Appl Med Care. 1995:91-5.
92 Canfield K. Priming intelligent split menus with text corpora for computerized patient record data-entry. Int J Biomed Comput. 1995 May 1;39(2):263-73.
93 Yamazaki S, Satomura Y, Suzuki T, Arai K, Honda M, Takabayashi K. The concept of "template" assisted electronic medical record. Medinfo. 1995;8 Pt 1:249-52.
94 Stephanidis C. Designing for interaction quality in health telematics. Stud Health Technol Inform. 2000;72:95-107.

Chapter 3

PropeR: a multi-disciplinary EPR system



Published as Helma van der Linden, Gerrit Boers, Huibert Tange, Jan Talmon, Arie Hasman, PropeR: a multi disciplinary EPR system, International Journal for Medical Informatics, Vol 70, 2003, p 149 – 160

PropeR: a multi disciplinary EPR system

1 Introduction

In the context of the PropeR project, an EPR system containing active guideline support needs to be developed for multi-disciplinary users. This system should be a distributed system, flexible enough to be useable in different domains. The system was built based on international standards such as the OMG HDTF specifications and the GEHR/*open*EHR archetypes, using available tools and techniques, such as the open source components of OpenEMed and the techniques of eXtreme Programming.

There are two cornerstones for our work: the Stroke Service which deals with the integrated care of stroke patients and the PropeR project that deals with the issues around the EPR system that supports the care process.

1.1 STROKE SERVICE

In 1996, a project started to develop and implement a new care model for the treatment of stroke patients in the Maastricht region of the Netherlands. [1] This model aims at making an assessment during the hospitalization after a stroke of the most appropriate discharge destination. One of the discharge options is to the patient's home environment, where they will receive rehabilitation treatments by a multidisciplinary team of primary care providers such as a speech therapist, a physical therapist, and an occupational therapist.

The first phase of the project involved setting up the stroke-service organization, achieving closer collaboration between the various disciplines and setting up new procedures. A recent review showed that the approach is promising: the costs are lower and patients are more satisfied [2].

The second phase has started in 2002 and involves the addition of guidelines to the integrated stroke-service model. The stroke-guidelines, published by the Netherlands Heart Foundation, are currently reviewed and adapted for use in the Stroke Service. [3]

1.2 PROPER PROJECT

PropeR is a project funded by the Netherlands Organisation for Health Research and Development. It studies the effect on the quality of care of decision support software integrated with an EPR. This project consists of two subprojects: one supporting the treatment of leukemia patients at the department of Internal Medicine

of the Academic Hospital Maastricht (azM) and one supporting the Stroke Service home care team. This article discusses the latter.
The Stroke Service EPR system supports the registration and retrieval of the data needed by the various disciplines and meets the communication needs of the team members. Electronic support of the usage of guidelines will be provided.

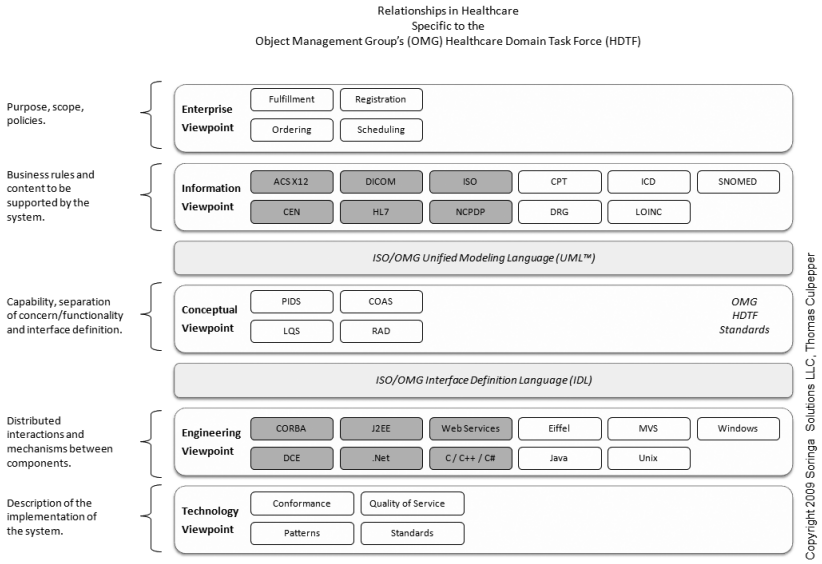
2 Requirements

The PropeR EPR should meet several requirements. These requirements are ranked from general to application specific requirements. Generally applicable requirements are simply requirements that hold for any ICT system in general (robustness, flexibility etc.) and for any medical information system in particular (like security and privacy).
The following requirements (from general to project specific) were specified:

- The software should be flexible enough to accommodate changing user requirements without a large programming effort.
- The EPR system should be robust, i.e. designed, implemented and tested according to state of the art software engineering methodologies.
- Current security standards should be implemented.
- Current privacy regulations should be adhered to.
- The software should be based on relevant national and/or international standards.
- The software should be generic enough to be usable in other domains and by other projects.
- The software should be open source.
- The software should be platform-independent.
- Data/software should be accessible from different locations.
- Data from different sources and disciplines should be handled.

We will further discuss the requirements of using (inter-)national standards in general and security standards in particular.
Over the years many standards have been developed. There are different kinds of standards, some define content, some define implementation, and some define communication with other systems.
Using relevant standards helps consolidate the future-proof aspect of a system as well as avoid reinventing the wheel. Standards also facilitate independent assessment of the software. Furthermore, standards allow for easier integration and interoperability with other systems, thus extending the usefulness of the software.

FIGURE 2 Reference Model for Open Distributed Processing with the OMG HDTF components [11]



Finally, software based on standards may be certified, thus enhancing its reliability and acceptance [4].

The system should be able to handle the clinical data used by the various disciplines involved in patient care. Combining this with the requirement that the system should be used in different domains and in different projects, and with the requirement that the system should be flexible enough to easily accommodate changing user requirements, we adopted the ideas of Beale to separate the specification of the domain knowledge from the implementation of the software. Ideally, by changing the specifications the application can be used in a different domain [5].

No medical system should be developed without any consideration for security. Security covers the aspects of authentication, access control, availability, integrity, accountability (non-repudiation a.o.), audit, and accuracy. Extensive research into security standardization has already been done, which allows us to benefit from it [6].

3 Design Considerations

We searched for an EPR system that would meet our requirements mentioned before and that could be extended with guideline support. We concluded that there was no such system available. This led to the decision to develop a system for the PropeR project that would be general enough for reuse in other projects.

The PropeR project team decided up front to try to develop a high quality EPR system while still avoiding reinventing the wheel as much as possible. To arrive at this goal, we tried to use as much of the tools, techniques, and standards already available that could be beneficial to the PropeR project.

3.1 STANDARDIZATION DEVELOPMENTS

3.1.1 Reference Model for Open Distributed Processing

We used the Reference Model for Open Distributed Processing (RM-ODP) (see Figure 2) as a checklist for our architecture. It is a generic framework developed by the ISO standardization institute. "The objective of ODP standardization is the development of standards that allow the benefits of distribution of information processing services to be realized in an environment of heterogeneous IT resources and multiple organizational domains." [7] The RM-ODP describes five viewpoints [8]:

1 Enterprise viewpoint: This viewpoint defines the purpose, scope and policies of the system.
2 Information viewpoint: This viewpoint is concerned with business rules and content to be supported by the system.
3 Computational viewpoint: This viewpoint is used to specify the functionality of a system in a distribution-transparent manner.
4 Engineering viewpoint: This viewpoint defines a model for distributed systems infrastructure.
5 Technology viewpoint: This viewpoint describes the implementation of the system and the information required for testing.

The model has proven its viability by the fact that it was developed in the mid 90s of the last century and is still in use, and it is frequently used as a basis for other models and architectures. Furthermore, it is developed by the ISO standardization institute, which prevents bias towards vendor-specific solutions.

3.1.2 OMG HDTF

The Object Management Group (OMG), founder of the CORBA specification, has expanded their working area to include domain-specific specifications. Their intention is to create specifications for (distributed) solutions to generic domain-related problems [9]. For the healthcare domain, a special taskforce (Healthcare Domain Task Force or HDTF) was formed [10]. The generic nature of the HDTF specifications allows them to be usable in other domains as well.

The OMG HDTF situated their components in the computational viewpoint (see Figure 2).
Below is a short description of the HDTF specifications that are used for the PropeR EPR. Other HDTF specifications are available or in progress, but will not be considered in the scope of this article.

3.1.2.1 *Person Identification Specification (PIDS)*

Until a few years ago, it was current practice in the medical domain to implement a medical information system with almost no connection to other medical systems. In general, hospitals or other larger medical institutions invested in the integration of various internal subsystems, but integration stopped at the boundary of the organization or the facility. This resulted in fragments of medical records in various systems each having their own identifier for the patient, often only known within the system itself. Creating an overview of all medical data on a person often failed because of the unavailability of the necessary identifiers or the uncertainty whether the IDs really belonged to the same person.
The HDTF tried to solve this problem by defining the Person Identification Specification (PIDS) [12]. The PIDS is a generic interface to a master patient index. It can handle multiple identifiers for a single person and also has functionality to dynamically correlate identifiers from multiple PIDS servers, thus providing the ability to build a patient identification service, serving an organization, a region or even a nation.
In general, a PIDS system can identify a person by means of a set of demographic and/or biometric parameters and return a list of IDs the person is identified with in various systems. This list can be filtered by authentication profiles, so the user can only retrieve the information he is authorized to see.
Beale and others recognize the PIDS specification as the de facto standard to the identification problem. [13]
In this article, the term PIDS server refers to a system, which consists of a database containing demographic information and an implementation of the PIDS specification that manages that information.

3.1.2.2 *Clinical Observation Access Specification (COAS)*

The medical domain requires context-rich data to enable the best possible decision on the medical care at hand. Current medical information systems almost all have their own proprietary terminology and format for storing data with or without elaborate context-related data, which makes it very difficult to compare data from different systems. Improvements have been made by adhering to communication standards in the form of messages, as specified in e.g. HL7. [14]

The HDTF developed the Clinical Observation Access Specification (COAS) [15] as a generic interface to different medical systems, which allows not only for the retrieval of context-rich data in a generic, thus exchangeable, format, but also adds extra functionality like queries into the future. The latter allows a user to request data, e.g. lab results, the moment the order for the data (i.e. the lab tests) is done. Once the data become available, the user is notified.
Analogous to the PIDS server the term COAS server will be used for any system containing clinical data that is interfaced through a COAS implementation.

3.1.2.3 *Lexicon Query Service/Terminology Query Service (TQS)*

No other domain has so many terms as the medical domain. It is said that almost 50% of the world terminology consists of medical concepts. [16] Numerous project teams have undertaken the quest to classify these terms into manageable sets, coding schemes and terminology lists to further enhance the comprehension of the medical data and indirectly contributing to the quality of care.
The Terminology Query Service (TQS), formerly known as Lexicon Query Service (LQS), [17] is an HDTF specification of a generic interface to terminology servers. The TQS offers functionality such as versioning, but also mapping of one coding scheme into another (provided the correct mapping data is available) and retrieval of specific information about a term, e.g. the text used for column headers or for patient information.
It is generally considered the most complex HDTF specification and full implementation will not be available in the very near future.

3.1.2.4 *Resource Access Decision Facility (RAD)*

The Resource Access Decision (RAD) [18] elaborates the standard CORBA security services and provides an interface to an authorization server to filter data based on authorization profiles. This allows for authorization at a very detailed level.
In a typical scenario a PIDS server or a COAS server returns the requested data to the user, but only after the RAD server has filtered out the data for which the user is not authorized.

3.1.3 OpenEMed

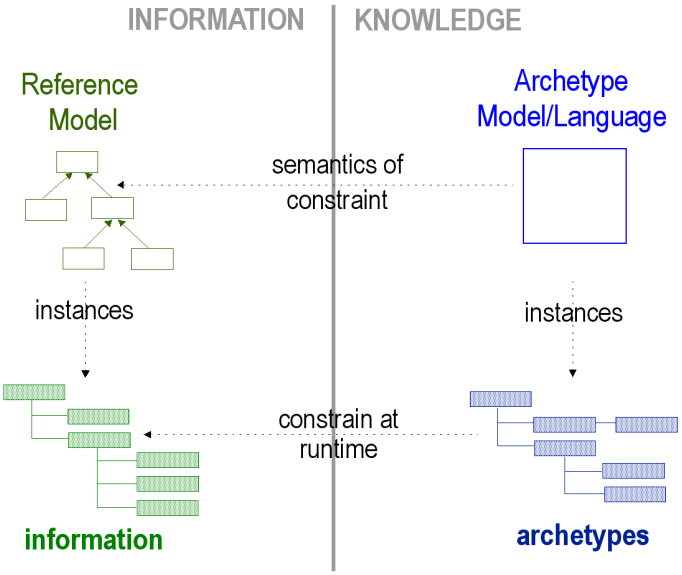
The OpenEMed team, resulting from the Telemed project [19], has made an open source, platform-independent implementation of parts of the OMG HDTF specifications, in Java.
This system is often considered as a reference implementation of the HDTF specifications. It provides a stable PIDS implementation as well as a functional COAS implementation. The latter extends the HDTF specification with data storage functionality.

It should be noted that the OpenEMed software is **not** a complete EPR system, but rather provides a collection of components to build such a system.
The OpenEMed team focuses its current activities on the implementation of the RAD specification.

3.1.4 GEHR/openEHR

One of the drawbacks of conventional system development (the data model/data definition is stored in the DBMS) is the fact that changes in the data model require changes in the database and often in the rest of the software. Medical information systems are often concerned with a large set of data, which may change due to newly gained insight or differences in viewpoints. On top of this comes the problem of hard coding complex domain-related concepts and their relations into the software by software developers. It is therefore crucial to find a solution to the problem of a frequently changing data model.
The team members of the first GEHR project (Good European Health Record) tried to solve this problem by moving the data definition to a separate component. This concept is elaborated further by the team of the second GeHR project (Good Electronic Health Record). The latter is succeeded by the *open*EHR initiative [20-22].
The data model is divided into two parts: a static part, which hardly changes over time, and a more dynamic part, which is more susceptible to changes in the medical knowledge.
Separating the dynamic part of the data model from the database, making it a separate component, will largely eliminate the proliferation of changes in the data model into the software.
*open*EHR uses the term archetype to describe a building block that defines a medical concept in a standardized, machine-understandable way. An archetype not only describes the structure of the concept, but also includes data validation rules. Archetypes can be adjusted to handle the changes in medical knowledge. All archetypes together form the dynamic part of the data model.
This construction leaves the knowledge maintenance largely to the domain experts, rather than forcing it onto the software engineers.
The archetype architecture of *open*EHR consists of four (object-oriented) models. (See Figure 3).
A **Reference Model**, that only defines generic objects that have a more or less static meaning and a long lifetime (e.g. "person", "event"), provides a stable model that will be valid longer than the usual IT-lifecycle of 3 to 5 years. Building software that implements this Reference Model provides a generic, future-proof system that can keep up with the change rate of the domain. The Reference Model is further specified and constrained by the **Archetype Model/Language**; a conceptual language

FIGURE 3 Four-model architecture from Beale [23]



that provides the basis for the archetypes (implemented in an archetype editor). **Archetypes**, in turn, are used by the system to generate objects that comply with the Reference Model. The data model (marked **information** in Figure 3) represents the system that stores the objects generated by the archetypes.
This four-model approach separates the development process in two parallel processes: the software development and the archetype development.

In summary, the Reference Model is implemented in software, so any object that is consistent with the Reference Model, can be handled by the software.
The creation of the data objects is done by a Validator. This is a software component that can read an archetype at runtime and use it as an input template for user input. Since an archetype contains validation rules, the input is validated and subsequently an object is generated with the structure defined by the archetype. This object contains the validated data and conforms to the Reference Model.
A very simple example should explain this.
Given a blood pressure, that consists of two values, the systolic value and the diastolic value, we will describe what kind of information can be found in each of these four model/architectures.
A specific blood pressure measurement is stored as a BloodPressure object that holds some administrative information (e.g. timestamp of measurement, timestamp of recording etc.) as well as two values A and B with a defined unit, that

comply with a given range and the rule that one is less than the other. The structure of this object as well as the constraint rule is defined in a BloodPressure archetype. The archetype, in turn, is composed using elements of the Archetype Model/Language, e.g. UNIT, QUANTITY, RANGE, RULE. The BloodPressure object is an instance of the Observation class that is defined in the Reference Model.
Archetypes can support the correct interpretation of information objects transferred from one system to another. This requires the need for standardized domain concepts. For maintenance reasons these concepts should be defined in ontologies the system can refer to.

3.2 ENGINEERING DEVELOPMENTS

3.2.1 eXtreme Programming

eXtreme Programming (XP) is a "lightweight methodology for small-to-medium-sized teams developing software in the face of vague or rapidly changing requirements" [24]. It focuses on delivering the best possible software based on the current requirements with the available resources (both money and people). It is very suitable for research projects where not all requirements are fixed or known beforehand and where resources usually are limited.
Although much of the requirements and design can be done before coding, the requirements or design often change while the coding takes place. XP therefore advocates iterations of design, testing and coding, with each iteration leading to a revision with enough functionality to make it cost-effective.
Without going into full detail, some of the XP techniques used in PropeR will be described here.

3.2.1.1 *Testing before coding*

Conventional software development methodologies (such as RAD, DSM/waterfall) include a significant testing phase at the end of the development process. The fact that errors of any kind are not revealed until very late in the project is a major disadvantage. XP tries to overcome this by the "testing before coding" technique. XP tests are written before a single line of actual code is written. XP distinguishes two levels of testing. Unit tests test the correct behavior of a unit, usually a class or closely related set of classes, and are usually written by a programmer. Functional tests test a required functionality of the system, usually implemented in one or more components. Ideally, the latter are written by a representative of the "customer", i.e. the future users, but usually they are aided by a programmer. All tests are usually written in the same programming language as the actual code and should be run automatically and constantly.

Since the tests are written, and run, before the actual code is written, they can also serve as a measurement for the progress made: the number of failing tests should decrease as the project progresses.

3.2.1.2 *The simplest thing that could possibly work*

Continuous testing allows for writing the most basic software that makes a test run correctly. XP holds the view that adding in extra features "in case they might be useful later" is a waste of time and money.
If requirements change, so do some of the tests. Only if the actual production code fails the new version of the test, it should be changed.

3.2.1.3 *Refactoring*

Changing requirements and gained insight over time can reveal that some parts of the software are too complicated or not well designed. Refactoring is substantially rewriting the code to result in better-designed or simpler code, while still complying with the tests. Since the tests serve as a safeguard against side effects, refactoring can be done throughout the entire development phase.
Refactoring is aimed at quality improvement, rather than bug fixing. The latter is often the only substantial rewrite of code in conventional methodologies.

3.2.1.4 *Multiple revisions*

Continuous testing should also be done on the entire system, to reveal any errors resulting from integration of the various components. This simplifies the release of many revisions: as soon as a user requirement is implemented and all tests pass, the software is stable enough to be released for further review by the customer. Users get the chance of reviewing the software very early in the process and comments and changes can be integrated in the development at a much lesser cost and quicker turnaround time.

3.2.2 Open source

Software reuse is a cost efficient way of developing software. Peer review of software code will increase its quality and functionality. These are the main reasons for the open source community to publish software code.
Every open source project is published under a license. The conditions of this license can vary, but all describe the conditions that apply when the code is modified, extended or reused. [25]
Usually the cost for end users of open source is far less than that of commercial software, but there might still be charges for obtaining the software or for services related to the software (e.g. documentation or support). To indicate that OS does

not necessarily have a zero price tag, the following comparison is often made: open software is free, not free as in "free beer", but free as in "free speech". [26]

4 Methods

Given all the techniques, components and tools described before, we will now describe the methods we have used to arrive at the architecture of our system. The architecture will be described in more detail in the next section.

To analyze and document the functionality of the PropeR system we used UML use cases, which we presented to the future users for comment. Based on these use cases, functional tests were written.

The RM-ODP model was used to check whether all viewpoints were taken into account and to reveal the missing pieces.

We decided to use the HDTF specifications because of their generic nature (i.e. not necessarily domain-specific) as well as their precise specification for implementation. The interfaces are specified in IDL (a platform and programming language independent specification language), which ensures that all servers can be accessed in the same way, returning the same results independent of the underlying implementation. An extra benefit is the fact that clients can be written without the servers even being present.

The HDTF specifications do not include any description of a generic data model, nor does the OpenEMed system. The *open*EHR architecture defines this missing piece in a way that meets our requirements (easy accommodation to user changes, portable to other domains).

One of the objectives of the PropeR project is to study the influence of active guideline support in combination with an EPR on the quality of care. The EPR should therefore contain a DSS component that can handle guidelines. The authors intend to integrate Gaston, [27] into the PropeR EPR system. Gaston is a knowledge inference engine that can generate reminders based on knowledge rules that are applied to provided medical data.

Both the archetypes and Gaston need to use an ontology for term reference. We intend to use Protégé, an ontology editor [28], to build a common ontology that can be shared by the (EPR) archetypes and Gaston.

The PropeR project will be written as open source. Both OpenEMed and Protégé are open source systems and both are increasingly using additional open source components as well.

The choice of the programming language was not a major issue, but the fact that OpenEMed is written in Java, tipped the scale in favor of this language.

As much unit tests as possible were written. Code refactoring was done when necessary.

5 Results

5.1 THE PROPER ARCHITECTURE

This section describes the PropeR architecture, as seen from the various viewpoints of the RM-ODP.

5.1.1 Enterprise viewpoint

The PropeR system is a multi disciplinary EPR system for use in a primary care environment. The users, in particular the care providers, belong to different organizations, and cooperate in a care team focused on patients. Team composition differs per patient, thus the care providers treat a different, but overlapping, set of patients. The PropeR system should be able to retrieve data from different sources as well as store information provided by the team. This information is shared between the team members. We will refer to this information as communication information, although it might well contain medical information, e.g. a discharge form that can be seen by all members. Three major functionalities can be distinguished:

- Patient identification, to identify the patient using a set of demographic parameters.
- Messages and medical information, to retrieve and store medical data relevant to the patient, supporting the communication between care providers in the stroke team. Since it should be transparent to the EPR system users (i.e. the team members) whether they enter medical information for their own purpose only or for communication with other team members, we view the storage and retrieval of this information as one functionality. The access to the different kinds of data should be handled by authorization profiles.
- Decision support, to support the use of guidelines by the stroke team.

5.1.2 Information viewpoint

The information viewpoint focuses on the business rules and content to be supported by the system. Models like HL7, CEN 13606, and various terminologies and coding schemes (e.g. Loinc and Snomed) provide the (semantic) context. Archetypes, providing the (syntax) rules, are also part of the information viewpoint.

The PropeR project anticipates the use of general standards and considers the use of the International Classification of Functioning, Disability and Health (ICF, formerly known as ICIDH) [29] as a candidate. The latter should be discussed with the us-

ers before adoption. The OpenEMed system has an American origin and therefore uses HL7 concepts. [14]
Guidelines for the team are also part of the information viewpoint.

5.1.3 Computational viewpoint

With the choice for the OMG HDTF specifications (PIDS, COAS, TQS and RAD), the computational viewpoint is described by these specifications.
Also part of the computational viewpoint is a decision support module engine that can handle the guidelines and a module that handles the archetypes.

5.1.4 Engineering viewpoint

The PropeR EPR system is a distributed, component based system with a multi-tier architecture using CORBA as middleware.

5.1.5 Technology viewpoint

Client/Server communication is done using the HTTPS protocol. XP tests are used to test the system.

5.2 IMPLEMENTATION

Focussing on the computational and engineering viewpoints, the architecture described in the previous section is implemented as a multi-tier architecture. We distinguish several layers (from serverside to clientside):

- Services layer, which includes the software that manages the underlying database. Whether this database is implemented in a separate layer is not relevant for this application.
- Control layer. This layer handles the interaction between users and servers.
- Presentation layer, which presents the information to the user.

Several services can be distinguished. All services consist of the information as well as the interface to the information.

- A demographic service, which handles demographic information as well as the query and the update functionality of this information.
- A medical service, that consists of three components:
  - A medical information component, which handles the storage and retrieval of clinical data and communication data as well as the modification of this information.
  - A component that can generate clinical data based on an archetype and user input. To stay in sync with Thomas Beale, we call this component the Validator.

    - An Archetype Server component that manages the repository of archetypes.
- A decision support module, which can handle the guidelines.
- A terminological service, which handles the references to the various terms and ontologies used by the system.
- A client module, which handles interaction with the user.

We will discuss the status of each of these components or modules in more detail.

5.2.1 Demographic module

The demographic module is already finished. It is based on the OpenEMed PIDSserver. We installed two servers, one to handle the patient information, one to handle the care provider information. This will increase privacy. These PIDS servers differ only in name and data, not in software code.

5.2.2 Medical module

The medical module is under development. The medical information component is based on the OpenEMed COAS server. Currently it is possible to store and retrieve COAS structures in a COAS server.

The Validator is currently under development. The problems encountered will be discussed later.

The Archetype Server component handles storage and retrieval of the archetypes and can handle versioning. Since archetypes represent medical concepts and archetype management requires versioning (among others), we consider the Archetype Server to be a specialized kind of terminology server. It is therefore logical to base a full-fledged Archetype Server on the HDTF TQS specification. This module has yet to be developed.

The archetypes are under development. It is not feasible within the context of the Proper project to build a full-fledged Archetype Editor, so we decided to manually construct the archetypes. This will be discussed later in more detail.

5.2.3 Decision Support module

The PropeR project intends to implement the most relevant guidelines for stroke rehabilitation in a machine-interpretable form. The Decision Support module will consist of Gaston and a repository of guidelines, in a format readable by Gaston.

Since Gaston is more or less a stand-alone system, a generic interface component has to be developed to create interoperability between Gaston and the PropeR EPR system.

Both the DSS module and the archetypes need a common ontology to build the

respective guidelines and archetypes. Therefore, we postponed the implementation of the DSS module until the archetypes are implemented. Furthermore, the goal of the PropeR project is to study the effect of the DSS on the quality of care. Since currently the participants in the project do not use any form of EPR system to support their combined efforts, it is necessary to introduce the PropeR EPR first without any DSS module.

5.2.4 Terminological module

The terminological module will consist of a server based on the HDTF TQS specification and the underlying information: an ontology based on the relevant coding schemes.
Sharing a common ontology simplifies the interaction between archetypes and Gaston, which makes it evident to develop a single common ontology.
Since there is no functional TQS server available and full implementation is beyond the resources of the project, a rudimentary substitute will be developed.

5.2.5 Client module

Although the initial idea was a Java application, we changed this to a web application with Java Server Pages and servlets. This would minimize problems resulting from installing software and improve the acceptance by the users.
This module is developed parallel to the development of the archetypes.

5.2.6 Security

The OpenEMed team is currently focusing on the implementation of the RAD specification [30]. This allowed us to postpone the implementation of specific security issues to a later phase. Basic security needs were satisfied by running the back end servers on protected machines and by running the web application over secure HTTP.

5.3 EXPERIENCES

We studied the use of UML and the HDTF specifications and used the resulting knowledge to model the information we retrieved from the encounters with the future users. Instead of building a class diagram containing all the relevant data items in a proprietary data model, we modeled the diagram in accordance with the COAS specification.

We designed separate use cases for the management of demographic information and medical information.

We also built screen mockups using simple drawing software, to get feedback from the users. This was mainly used to check the completeness of the data set.
From that point on most of the effort went into understanding the OpenEMed code and writing the PropeR related code. A visit to the OpenEMed team was very instrumental to a mutual understanding of the OpenEMed and the PropeR project. The OpenEMed team agreed to participate in the modification of the OpenEMed software should this be needed by the PropeR project, while the PropeR team agreed to contribute modifications to the OpenEMed system.
Following the XP technique of many releases, we conferred with the users to decide on the most useful functionality for the first version. Where possible, unit tests for the PropeR code were written.

We installed the OpenEMed PIDS and COAS servers on various PCs with either Windows 2000 or Red Hat Linux and thus checked platform independency.
Tests showed that this cross-platform implementation caused no problems. We also successfully experimented with a connection from Dublin, Ireland to our locally installed PIDS server.
We further complicated the implementation by successfully running the PIDS servers as well as the client on one CORBA ORB (OpenORB), and the COAS server on another (Orbacus), with all of them connecting to an Orbacus NameService. This effectively proved ORB independency.
While writing the software code, we also experimented with the fault tolerance of the system. The system proved to be very stable. It was not possible to make the system crash by feeding invalid data structures. Invalid structures invoked the respective server to write error messages to the console and the respective log file, but subsequent feed of valid structures was handled correctly without any human interference.
Experiments also proved that it was possible to terminate a running server and restarting it (either in the same location or on an entirely different machine) without terminating the session with the client.

6 Problems encountered

Making the choices explained before (standards, open source software) contributed to the idea that the projected outcome (high quality software) was feasible within the timeframe of the project. Everyday experiences however showed us that it was definitely not a mere jump onto a running train.

6.1 OPEN SOURCE

The PropeR team is of the opinion that the software produced by the project should be released as open source. The project is funded with public money, which makes it fair to publish the results to the community for reuse. Moreover, comments on the code that could improve the quality are greatly appreciated.
Using open source software has great benefits. It allows a jumpstart and it is possible to modify the software to the specific needs of a project. There are, however, some drawbacks. Usually the software is written to meet the needs of the original authors and it takes time and effort to discover missing parts that are unnecessary in the original environment, but necessary in the project at hand.
This was also experienced in the PropeR project. It turned out that the original PIDS implementation did not support composite traits[2], which was necessary to implement the relationship between a patient and the care providers that formed his care team. The teams from PropeR and OpenEMed worked together to implement a solution for this problem. It is now part of the current version of OpenEMed.

6.2 ARCHETYPES

As mentioned before, Archetypes are based on an Archetype Model/Language, to force consistency between archetypes and conformance to the Reference Model. The *open*EHR team chose to implement this Archetype Model/Language into an Archetype Editor. Current developments in the *open*EHR project lead to a revision of their Reference Model and the archetype editor based on that. These revisions will not be available before the end date of the PropeR project. Furthermore, the PropeR archetypes should generate COAS objects, which differ from the *open*EHR objects. All this led to the conclusion that we could not use the *open*EHR archetype editor.
It is beyond the scope of the PropeR project to build a custom-made Archetype Editor based on the PropeR Reference Model, thus the archetypes will be created by hand.

We try to implement the archetypes as simple as possible, but in such a way that the EPR software does not need to have knowledge of the data concepts used.
We defined a general COAS structure in XML Schema. This can be viewed as an XML Schema representation of the Reference Model.
Archetypes will then be hand coded in XML and validated against the COAS XML Schema. This will still generate COAS objects that are validated against an XML Schema.

2 A simple trait is a term used in the PIDS specification to describe a key/value pair. A composite trait is a key/value pair, where the value can consist of one or more traits.

6.3 SCREEN REPRESENTATION

The conventional way of designing a user interface consists of carefully designed screens or dialogs where each widget on the screen is more or less one to one connected to a specific data structure. This will lead to ergonomic and pleasing user interfaces. The down side to this approach is the necessity to change the user interface when the (conventional) data model changes.

An alternative is the generation of screens based on a generic layout, usually a list of label/value pairs. This approach accommodates data model changes very well, but will usually not result in attractive screens.

Archetypes describe data structures that can be used as a stand-alone concept, or a hierarchy of structures and substructures. Archetypes move the changes in the data model away from the database to the archetype repository. Adding a screen representation to an archetype seems convenient, but also has some drawbacks. The first issue to be considered is the fact that dedicated clients can be used to display (subsets of) data in a different way, e.g. a web based client, a stand-alone application or a PDA-client. This would result in many different screen representations connected to a single archetype. Furthermore, a new type of client would necessitate adding a new screen representation to the archetype.

Another issue deals with the possibility of representing data, especially numeric data, in various forms, like lists, graphs etc. This would require screen representations for all kinds of data representations.

These considerations lead to the conclusion that it is more appropriate to store screen representations separate from the archetypes, but that would mean that an update to an archetype might require an update of a separate repository of screen representations.

Therefore, some more study into this problem has to be done to find the optimal compromise between handcrafted ergonomically attractive screens and less attractive generated screens, which can solve the issues addressed before.

7 Discussion

We end with a discussion of the advantages and disadvantages of the tools and techniques used to achieve the PropeR EPR.

Fitting together the bits and pieces to arrive at a component-based EPR based on standards, is not a mere copy and paste operation. All pieces have to be studied closely to be able to decide if modification is necessary to make them fit and to discover which pieces are missing.

Many believe that building software based on standards will greatly benefit the quality of the software and its functionality with respect to the communication and interoperability with other software systems. Our experience is that building an EPR system based on standards is not a trivial thing to do. Although standards are available, not all of them are ready for use. Some are still under development and many are not yet implemented in software. Furthermore, there are (too) many standards to choose from.

The challenge of building a standardized EPR system turned out to be very time consuming but interesting and beneficial enough to pursue to the end. We are more than ever convinced that this will result in an EPR system that has a life cycle beyond the PropeR project and that will be easily adaptable to other domains due to the archetype approach.

Using open source has proven to be timesaving. It would never have been possible to write software with the same functionality and quality of the OpenEMed system with the limited resources of the PropeR project. Discussions with the OpenEMed authors also improved the quality of the software of the OpenEMed system as well as of the system under development for the PropeR project.

Using the eXtreme Programming technique of writing unit tests has also proven to contribute to the quality of the software. Focusing on writing all kinds of tests generally increases the number of tests that will be performed. Since these tests will be written once and executed many times, there is no objection against writing as many tests as possible. Running the tests often, i.e. always after the implementation of a coherent part of the production code, quickly reveals side effects that occur when code is changed to accommodate different needs. This makes it much easier, i.e. more cost efficient, to trace the origin of these side effects and remove them.

We mentioned already the need for a domain ontology to increase the portability of archetypes across systems and care providers. The portability issue is further complicated by the fact that various care providers can use localized terms (i.e. translated into the local language or to discipline dependent terms). It is therefore necessary to develop or use either a language independent ontology or an ontology that can map the coded terms to the various local synonyms/translations. In either case, a domain ontology is necessary that will be accepted by a representative group of different medical specialists.

The team members of the Stroke Service still use conventional means of communication like paper, telephone and fax. After implementation of the first version of the PropeR EPR system, we hope the communication and cooperation will improve. The information will be available at the time and place most convenient for the care provider. Rather than receiving e.g. a lengthy telephone call with all the

necessary information in the midst of a therapy session, it will be sufficient to notify the team member of a new patient and the therapist can look up the information at a more convenient time.

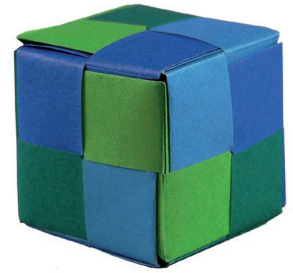
8 References

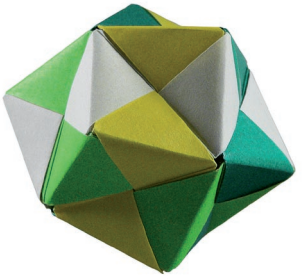
1 Transmuraal Zorgmodel CVA Regio Heuvelland. Heerlen: Synchron; 1996 april 1996.
2 Beusmans G, Zutphen Hv. Invitational Meeting Transmuraal Zorgmodel CVA Regio Maastricht en Heuvelland. Maastricht: AZM; 2002 april 2002.
3 Commissie CVA-revalidatie. Revalidatie na een beroerte, Richtlijnen en aanbevelingen voor zorgverleners. 2002 [cited 2002 December 2002]; Available from: http://www.hartstichting.nl/Uploads/Brochures/mID_5641_cID_4622_revalnaber.pdf
4 Kokolakis S, Gritzalis D, Katsikas S. Why we need Standardisation in Healthcare Security. In: Allaert F, Blobel B, Louwerse K, Barber B, editors. Security Standards for Healthcare Information Systems. Amsterdam: IOS Press; 2002. p. 7 - 11.
5 Beale T, Goodchild A, Heard S. Design Principles for the EHR. 2002 1 april 2002 [cited 2002 4 November 2002]; 2.4: Available from: http://www.openehr.org/downloads/design_principles_2_4.pdf
6 Blobel B. Analysis, Design and Implementation of Secure and Interoperable Distributed Health Information Systems. Amsterdam: IOS Press; 2002.
7 ITU-T X.901 | ISO/IEC 10746-1 ODP Reference Model Part 1 Overview. Strathfield, NSW, Australia: ISO/IEC; 1995.
8 Raymond K. Reference Model of Open Distributed Processing (RM-ODP): Introduction. [cited January 2003]; Available from: http://archive.dstc.edu.au/AU/research_news/odp/ref_model/papers/icodp95.ps
9 Object Management Group. [cited December 2002]; Available from: http://www.omg.org
10 OMG Healthcare DTF. [cited January 2003]; Available from: http://www.omg.org/healthcare
11 Culpepper TC, Brinson T. Reference Model for Open Distributed Processing (RM-ODP). 1999 [cited January 2003]; Available from: http://cgi.omg.org/docs/health/01-04-04.pdf
12 Person Identification Service (PIDS) Specification. 2001 [cited December 2002]; Version 1.1: Available from: http://cgi.omg.org/docs/formal/01-04-04.pdf
13 Beale T. Health Information Standards Manifesto. 2001 [cited January 2003]; Revision 2.5: Available from: http://www.deepthought.com.au/health/HIS_manifesto/his_manifesto.pdf
14 Health Level Seven Implementation Support Guide for HL7 Standard Version 2.3. 1998 [cited 2003 January 2003]; Available from: http://www.hl7.org/Special/IG/final.pdf
15 Clinical Observations Access Service Specification. 2001 [cited December 2002]; Version 1.0: Available from: http://cgi.omg.org/docs/formal/01-04-06.pdf
16 Baud RH, Lovis C, Rassinoux AM, Sherrer JR. Alternative ways for knowledge collection, indexing and robust language retrieval. Methods of Information in Medicine. 1998 November 1998:315-26.
17 Lexicon Query Service Specification. 2000 [cited December 2002]; Version 1.0: Available from: http://cgi.omg.org/docs/formal/00-06-31.pdf
18 Resource Access Decision Facility Specification. 2001 [cited December 2002]; Version 1.0: Available from: http://cgi.omg.org/docs/formal/01-04-01.pdf
19 OpenEMed. [cited December 2002]; Available from: http://www.openemed.org
20 Good European Health Record London. [cited December 2002]; Available from: http://www.chime.ucl.ac.uk/work-areas/ehrs/GEHR/
21 Beale T, et al. Good Electronic Health Record Australia. [cited January 2003]; Available from: http://www.gehr.org
22 OpenEHR. [cited 4 November 2002]; Available from: http://www.openehr.org

23 Beale T. Archetypes: Constraint-based Domain Models for Future-proof Information Systems. OOPSLA 2002; 2002; 2002.
24 Beck K. Extreme programming explained: embrace change. Third printing ed: Addison-Wesley; 2000.
25 The approved licenses. [cited January 2003 27 January 2003]; Available from: http://www.opensource.org/licenses/index.php
26 The Free Software Definition. [cited 27 January 2003]; Available from: http://www.gnu.org/philosophy/free-sw.html
27 de Clercq PA, Hasman A, Blom JA, Korsten HHM. Design and implementation of a framework to support the development of clinical guidelines. Int J Med Inform. 2001;64:285-318.
28 Grosso WE, Eriksson H, Fergerson RW, Gennari JH, Tu SW, Musen MA. Knowledge Modeling at the Millennium (The Design and Evolution of Protege-2000): Standford University; 1999.
29 International Classification of Functioning, Disability and Health. 2001 [cited January 2003]; Available from: http://www3.who.int/icf/onlinebrowser/icf.cfm
30 Visit to Los Alamos National Laboratory. Los Alamos National Laboratory; July 2001.

Chapter 4

PropeR revisited





Published as Helma van der Linden, Huibert Tange, Jane Grimson, Jan Talmon, Arie Hasman, PropeR revisited, International Journal for Medical Informatics, Vol 74, 2005, p 235 – 244

PropeR revisited

1 Introduction

The PropeR project in the Maastricht region of the Netherlands [1] studies the effect of a combined electronic health record (EHR) and decision support system on the quality of care. The research is conducted in two settings. One of the settings is a primary care setting where stroke patients receive rehabilitation therapy in their own home from a multi-disciplinary team of primary care practitioners (PropeR Transmuraal).

The stroke rehabilitation team currently uses conventional communication methods like paper forms, phone calls, and fax messages. In this setting, the role of the EHR system to facilitate communication is studied.

The other domain is the hematology department in the University Hospital Maastricht. In this domain the focus is on the proactive protocol based support of clinicians. The PropeR EHR system (PropeRWeb) will be used in both domains.

In a previous paper [2] (chapter 3), we presented the architecture of the system. In this paper, we present the implementation of this architecture in the EHR system for stroke rehabilitation.

2 Stroke rehabilitation

The Maastricht Stroke Project provides a special care path for stroke patients. Once admitted to the emergency department, stroke patients are transferred to a medium-care stroke unit as soon as possible. At the stroke unit, the patients are stabilized and a series of diagnostic procedures are conducted. Based on the results of these diagnostics, a special stroke team decides upon a discharge destination for the patient.

One of these destinations is rehabilitation at home. Patient care is provided by a primary care stroke rehabilitation team consisting of a physical therapist, a speech therapist, and a community nurse. Home rehabilitation is coordinated by a home-care stroke coordinator. Discharge information from the hospital is disseminated in different ways. Medical and nursing discharge letters are sent via the stroke coordinator. Paramedical discharge information is communicated peer to peer. The members of the rehabilitation team have regular meetings to discuss the progress of their patients and to adjust their treatment. Between two meetings, team members communicate through telephone, fax, email, or a paper-based patient

record that resides at the patient's home. The communication between hospital and rehabilitation team and between rehabilitation team members is not optimal in the home rehabilitation. This is the reason for developing the PropeRWeb stroke EHR system.

3 Design considerations

The requirements and the architecture of this EHR system are extensively discussed in chapter 3. We will only summarize this information here.

3.1 REQUIREMENTS

Flexibility was the most prominent requirement for the PropeRWeb EHR system. We had several reasons for this:

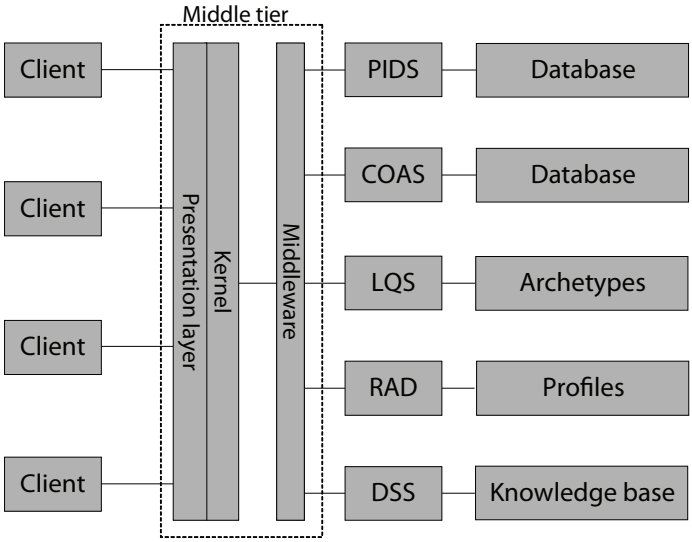
- The system had to accommodate two very different domains (stroke and hematology).
- The system had to be customizable to changing user needs.
- The system had to be compliant to different standards.

Additionally, we wanted to use open source components, not only because of the possibility to modify the software according to our needs, but also because open source components are believed to be more robust. Finally, the software should have a life cycle that extends beyond the PropeR project.
Several standards in healthcare exist. We state here the standards that we wish to comply to. The implementation of these standards will be discussed in detail later.

- OMG HDTF — a set of standards that are defined to enhance privacy, security and interoperability in a distributed environment [3-7].
- HL7 — a set of standards for messaging and data definition to enhance interoperability [8, 9].
- Although not specific healthcare standards, web-related standards like HTTP and SSL allow for secure access from different locations.

The future users of the PropeRWeb EHR differed significantly in computer experience. They also had to access the system from different locations, some of which did not allow installation of additional software. Hence, we chose for a web-based application. We also tried to simplify the learning curve by using screen forms similar to the current paper forms.

FIGURE 4 PropeR EHR architecture, see §3.2.1 for explanation of the acronyms



3.2 ARCHITECTURE

Figure 4 shows a schematic architecture of the PropeR EHR system. The left hand side is the front-end or user interface. The right hand side is the back-end or server-side. The system is highly component-based. We will briefly discuss the various components from right to left.

3.2.1 Servers

Each server consists of a database with an interface layer. This interface layer is responsible for accessing the underlying data and presenting it in a generic structure, thus removing the differences in data layout and data structure between various data sources.

The top four servers have an interface based on the OMG Healthcare Domain Taskforce (HDTF) specifications: from top to bottom the Person Identification Specification (PIDS) [5], the Clinical Observation Access Specification (COAS) [7], the Lexicon Query Specification (LQS) [4] and the Resource Access Decision Framework (RAD) [6], respectively.

Although there is no OMG HDTF specification for a Decision Support Service (DSS), in Figure 4 this component is represented in a similar way as the other services. All servers will be discussed in detail in § 4.1.

3.2.2 Middle tier

The middle tier consists of an object request broker (ORB), an application kernel

and a presentation layer[3]. The use of an ORB allows for separation of servers and front-end applications, thus enhancing security and interoperability.
We define application logic as all general-purpose functionality for e.g. connections to the database and conversions of the data for presentation purposes.
Domain knowledge comprises the knowledge about the domain concepts, e.g. drugs, lab tests and diagnosis, as well as the knowledge about business processes in which these concepts are involved (e.g. before issuing a prescription, contraindications and medical interactions should be checked).
We aim to separate the domain knowledge from the software as much as possible by containing it in loosely coupled components. Concepts are defined by means of archetypes and domain processes are defined e.g. in the rules of the DSS system.
Separating application logic and domain knowledge should minimize the number of changes in the software, when implementing changes in user requirements, business concepts or even a change of domain.
The kernel of the PropeRWeb system is responsible for the application logic, i.e. the connections to the servers, the conversion of the data for easier representation by the presentation layer and the audit of the data. The presentation layer is the interface to the client-side. It contains the logic to present the data produced by the kernel to the user and to handle user interaction.

3.2.3 Client-side

The client application should be simple to use from the user's perspective as well as from the systems administrator's perspective. We did not want to have patient data and application software stored locally, to avoid security vulnerability and synchronization problems. This resulted in the choice for a standard web browser as client application.

3.3 ARCHETYPES

In conventionally designed database applications, database tables and stored procedures reflect the application domain with its concepts and rules. Changes in the domain concepts or domain rules practically require changes in the database, which propel through the rest of the software. Switching domains even requires a complete rewrite of the database design, which makes it in fact a different application. To minimize the impact of such changes, the domain model has to be separated from the software and the database as much as possible. This sepa-

3 Although the client is responsible for the actual presentation to the user, we feel it is more appropriate to position the presentation layer in the middle tier, since all "knowledge" on the layout of the page and its content resides there. As discussed later, the client merely consists of a standard web browser.

ration can be established by using the concept of archetypes. Archetypes were first defined in the GEHR project and its successor the *open*EHR project [10, 11]. An Archetype can be described as a template[4] for a medical domain concept. A medical domain concept (which may vary from very detailed like "blood pressure" to very generic like "medical history") consists of elementary data structures. For example a data structure of AddressType consists of a street field, a postal code field, a city field and a country field and has an attribute that qualifies this address as "work address" or "most recent address". Every clinical data that is stored in the clinical database is an instantiation of an archetype. Archetypes are stored in an Archetype Repository, outside the clinical database. Domain concepts are thus separated from the software and the database, which makes the system robust against changes in medical concepts. Ideally, medical experts maintain the domain logic (the archetypes), while software engineers maintain the application logic and the clinical database, which contains only a generic set of data structures. Note that the user interface is still a joint effort of medical and software specialists.

4 Implementation issues

4.1 SERVERS

The PropeR EHR system separates demographic content from medical content, thus enhancing privacy, by using PIDS and COAS. The PIDS and COAS specifications were already implemented in the OpenEMed project [12], which is commonly regarded as a reference implementation and available as open source.

These server-side components were combined with an adjusted version of archetypes and an application kernel that controls the data flow. The various components will be discussed in detail below.

4.1.1 PIDS in the PropeR EHR

The demographic information of a person has been implemented as a PIDS identity. An Identity is a data set (a profile) that uniquely defines a person in one particular domain, combined with an ID. The elements of a profile are defined as a hierarchical data structure called Trait. A simple Trait with only one value can be considered a key/value pair. A composite Trait consists of one or more Trait structures. Figure 5 shows an Identity with Traits, expressed in XML. Line 2 holds the internal ID, i.e.

4 There are many definitions for template. In the context of this article, we use template in a common definition of the fill-in-the-blanks form.

FIGURE 5 PIDS Identity with Traits

```
1   <Identity>
2       <Id>85432</Id>
3       <Trait Type="HL7" Name="PatientName">
4           <Value>Practitioner^Jim^^^^</Value>
5       </Trait>
6       <Trait Type="HL7" Name="PatientAddress">
7           <Value>UM-MI Postbus 616^^6200 MD^Maastricht^^Nederland^ </Value>
8       </Trait>
9       <Trait Type="HL7" Name="PhoneNumber_Home">
10          <Value>^email@mydomain.com^31^043^1234567^ </Value>
11      </Trait>
12      <Trait Type="S" Name="FaxNumber_Home">
13          <Value>^^31^043^1234568^</Value>
14      </Trait>
15      <Trait Type="NODE" Name="PatientRelation">
16          <Trait Type="NODE" Name="Relation">
17              <Trait Type="S" Name="QualifiedPersonId">
18                  <Value>DNS:proper.mi.unimaas.nl/930</Value>
19              </Trait>
20              <Trait Type="HL7" Name="Role">
21                  <Value>DEV</Value>
22              </Trait>
23              <Trait Type="HL7" Name="StartDate">
24                  <Value>20020101 </Value>
25              </Trait>
26              <Trait Type="HL7" Name="StopDate">
27                  <Value>20021201 </Value>
28              </Trait>
29          </Trait>
30      </Trait>
31  </Identity>
```


in the data source[5] of the PIDS server. This ID is unique in the domain. Lines 3 – 14 show various simple Traits. Each simple Trait has a type and a name (line 3) that uniquely define the trait. In this case HL7 Version 2.3 names are used, but it could easily be switched to CEN 13606 names. Line 4 shows the actual value of the trait. Since HL7 V2.3 requires substrings with ^ as delimiters, the values are thus formatted. Lines 15 – 29 show a Composite Trait, which contains Traits instead of a value.
Like any other EHR system the PropeRWeb EHR involves two different populations: patients and care practitioners. We separated both populations into two different PIDS servers. This would enhance security and privacy and would also give us the opportunity to study the behavior of a system with multiple PIDS servers. Furthermore, we can use the same setup in both environments (stroke and hematology). In the hematology environment, we use the patients PIDS server as a gateway to the hospital information system (HIS), the professionals PIDS server interfaces to a local database.

4.1.2 COAS in the PropeR EHR

The COAS specification provides generic access to a system holding medical data. The data structure defined by the COAS specification is similar to the Trait structure

5 Note that a "data source" can range from a simple database to a hospital information system, anything that holds the relevant data.

of the PIDS, i.e. it is also hierarchical in nature. However, the COAS data structure makes a distinction between data and qualifiers or metadata.
A COAS Observation Data structure (ObsData) consists of ObsQualifiers and either an ObsValue or one or more ObsData structures. ObsQualifiers contains the qualifiers, structured in one or more ObsData structures. ObsValue contains the actual value of an ObsData structure plus a type indication. Figure 6 shows a typical COAS data structure, describing a vaccine administration. Lines 3 – 16 show examples of ObsQualifiers, while lines 19 – 56 describe the vaccine used, its dosage and some notes.
The OMG COAS specification provides read-only access to the underlying medical data. This is sufficient for accessing legacy systems. The OpenEMed team extended this specification with functionality to store data in a local database. We used the OpenEMed COAS server without modification for the stroke rehabilitation EHR.
Several stroke team members use a specialized EHR system for their own discipline. Initially we wanted to use various COAS servers to accommodate a connection to these systems, allowing the users to enter data through their own system. Closer study of the data set of the multidisciplinary EHR showed that there was virtually no overlap with the data in the team members' systems. This allowed us to simplify the design of the PropeRWeb system to one COAS server.
In the hematology project, we will accommodate a second COAS server to integrate data from other systems in the hospital.

4.1.3 LQS in the PropeR EHR

The Lexicon Query Service (LQS) is not a terminology server in itself, but specifies a generic interface to a terminology server. The OMG specification team acknowledged that an LQS is a very elaborate specification, which will not likely be fully implemented ([4], pages 1 – 10). We found no current implementation of a LQS server with enough functionality and it was beyond the scope of this project to develop an LQS server ourselves. Hence, we implemented the basic functionality of an LQS server into the kernel and left the implementation of a real LQS server to the future.

4.1.4 Other servers

The other servers, RAD and DSS, are currently not yet implemented in PropeRWeb. The OpenEMed team is currently implementing the RAD specification [6]. We wait until this open source RAD is available.
The stroke environment currently has no need for a DSS component. It will be integrated in the implementation of the hematology system.

FIGURE 6 Structure of COAS ObsData, describing a vaccine administration, provided as sample data in the OpenEMed project. Some parts are removed due to space limits.

```
<ObsData Code="&Immunology;">
    <!-- ObservationQualifiers -->
    <ObsQualifiers>
        <ObsData Code="&PatientIDInternalID;">
            <ObsValue>
                <HL72.3 ty="CX"><![CDATA[10006^^^^^&OpenEMed.org&DNS]]></HL72.3>
            </ObsValue>
        </ObsData>
        <!-- Authorize attribute ObservationDate_Time -->
        <ObsData Code="&ObservationDate_Time;">
            <ObsValue>
                <HL72.3 ty="TS"><![CDATA[19990730152800MDT]]></HL72.3>
            </ObsValue>
        </ObsData>
<!-- … more qualifiers … -->
</ObsQualifiers>
    <!-- End of ObservationQualifiers -->
    <!-- Vaccine -->
    <ObsData Code="&Vaccine;">
        <!-- VaccineType -->
        <ObsData Code="&VaccineType;">
            <ObsValue>
                <CodedElement>
                    <QualifiedCodeStr>DNS:OpenEMed.org/TraitCode/Immunology/Vaccine/VaccineType
                    </QualifiedCodeStr>
                    <PreferredText>03</PreferredText>
                </CodedElement>
            </ObsValue>
        </ObsData>
        <!-- Manufacturer -->
        <ObsData Code="&VaccineManufacturer;">
            <ObsValue>
                <CodedElement>
                    <QualifiedCodeStr>DNS:OpenEMed.org/TraitCode/Immunology/Vaccine/VaccineManufacturer
                    </QualifiedCodeStr>
                    <PreferredText>Parke-Davis</PreferredText>
                </CodedElement>
            </ObsValue>
        </ObsData>
        <!-- ExpirationDate -->
        <ObsData Code="&VaccineExpirationDate;">
            <ObsValue>
                <TimeStamp>20010720180000MDT</TimeStamp>
            </ObsValue>
        </ObsData>
        <!-- LotNum  information -->
    </ObsData>
    <!-- End of Vaccine -->
    <!-- … Information on Dosage … -->
    <!-- Note -->
    <ObsData Code="&Note;">
        <ObsValue>
            <PlainText><![CDATA[Vaccine: measles, mumps and rubella virus vaccine]]></PlainText>
        </ObsValue>
    </ObsData>
</ObsData>
<!-- End Of Immunology -->
```


4.2 MIDDLE TIER

The middle tier consists of three parts: the presentation layer, the kernel and the middleware. We have tried to separate them as much as possible.

This tier has the following functionalities, separated into the three parts.

1 **Kernel and middleware**: Provide access to the servers (both PIDS and COAS), for storage and retrieval

2 **Kernel**:

- Handle archetypes, forms and their instantiations
- Facilitate audit trails for medical data

3 **Presentation layer**:

- Process the user actions
- Present the data to the user

We use the open source OpenORB (http://www.openorb.org) as middleware. Since this is a standard component which we use unmodified, we will not discuss it here. The first functionality is implemented partly in the kernel, partly in the middleware. The middleware serves as locator and gateway; the kernel decides which server and function to address.

4.2.1 Archetypes in the PropeR EHR

Archetypes are representations of medical concepts, equivalent to the way terms in a coding scheme represent medical concepts. Archetypes are implemented as templates and stored in an archetype repository. Any object that holds clinical information is an instantiation of an archetype. Like coding schemes, archetypes face the problems of different terminologies, synomyms, and version management. Studying the OMG specifications, we concluded that the Lexicon Query Service (see before) is the most appropriate interface to the Archetype Repository. However, since there was no open source LQS implementation available, we chose a temporary solution and appended the archetype repository directly to the application kernel. We defined archetypes in XML (Figure 7). Each archetype has a name (element code, lines 2 and 5) and a value type (e.g., TextType, DateType, ListType). Restrictions to possible values can be defined within the value type (lines 6 – 9).
By providing the domain of this name ("DNS:proper.mi.unimaas.nl/" in line 2 and 5 of Figure 7), terms become more easily mappable to their synonyms in other coding schemes, but also allow the use of terms of various coding schemes in one archetype. An example: "DNS:proper.mi.unimaas.nl/Visus/Quality" and "DNS:who.org/ICF/V1.0/b2102" [13] both refer to the quality of visual functions.[6]

FIGURE 7 PropeRWeb archetype definitions

```
1   <elements>
2       <element code="DNS:proper.mi.unimaas.nl/CVA/Type">
3           <TextType/>
4       </element>
5       <element code="DNS:proper.mi.unimaas.nl/CVA/Location">
6           <ListType>
7               <item code="LEFT"/>
8               <item code="RIGHT"/>
9           </ListType>
10      </element>
11      <!-- ...more definitions... ->
12  </elements>
```

6 In our implementation, we have used the terms in local use, rather than the terms from a standardized classification.

4.2.2 Kernel

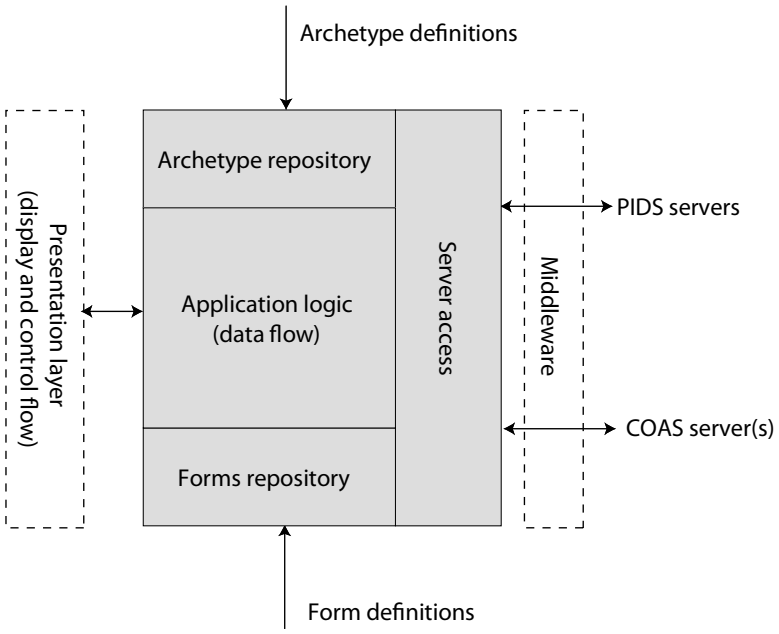
The kernel uses Java Beans and Java servlets for the data flow and Java Server Pages (JSP's) for display and control flow. The servlets and beans communicate with the PIDS and COAS servers through OpenORB. We used software of the open source project OpenEMed [12] as the basis for the application kernel of ProperWeb. We extended this kernel with a forms repository and an archetype repository, both defined in XML (Figure 8).

The server access part of the kernel is the only place in the ProperWeb system where the patient's ID from a PIDS server is linked to the appropriate ID in a COAS server. When more PIDS or COAS servers are linked to the ProperWeb system, we expect we have to change only a few Java classes.

We define auditing as a functionality that provides info on the existence and value of data at a specific moment in time, i.e. it must be possible to reconstruct the set of data as seen by a specific user at a specific time. Auditing is implemented in PropeRWeb by distiguishing between versions of instances of both archetypes and forms. An instance is identified by an instanceID; a specific version of an instance is identified by the instanceID *and* a timestamp of recording in the database. Each version of each instance of both forms and archetypes are stored in the database. The following examples should clarify this.

A user creates a new form and adds new information. The software creates a new instance of the form definition as well as new instances of each of the archetypes

FIGURE 8 Kernel components



for which information is filled in. The archetype instances are stored in the database, which provides the timestamp for the version of each instance. The form instance is updated to include references to the archetype instanceIDs and their version timestamps. Finally the form instance is stored with a versioning timestamp.
A user retrieves this form and modifies it by adding content as well as modifying existing content. The software creates a new version of this form instance (i.e. the instanceID is equal, but when stored, the database adds a new timestamp). New archetype instances for the added content are created and new versions for the modified content are created. The archetype instances are stored and the database provides the timestamps for the version. The version of the form instance is updated to include the references to the resulting set of archetype instanceIDs and their versions. Unmodified content is referenced by the same instanceID and version timestamp as in the previous version of the form.
Additionally, logfiles are kept that can also provide audit information.

4.2.3 Presentation layer

The presentation layer consists of Java Server Pages and two XSL stylesheets. This layer is fed by the application logic part of the kernel with XML representations of clinical data, archetypes and screen forms. Java Server Pages using XSL stylesheets transform this XML to HTML pages and extend these pages with some static elements such as menu, header and footer. This is implemented with a Resin webserver/servlet engine (http://www.caucho.com).

4.3 CLIENT-SIDE

Given the requirements and the architecture described above, the PropeRWeb system presents itself to the users as an electronic version of the familiar paper forms. This is implemented in a web application that asks the user to select a patient and subsequently presents a list of available medical forms. The user can select a form, which can be viewed, created or modified. If multiple instances of a form exist, a list is shown and the user can select one particular instance. This introduced the concepts of Forms and Form Instances in the system. A Form represents a template, i.e. the paper form, whereas a Form Instance contains the actual data of that form, i.e. a paper form filled in.
To avoid a local cache of sensitive data, we used non-persistent cookies, which are automatically deleted at the end of the session.

5 Discussion

5.1 STANDARDS

Several standards were implemented. The OMG HDTF specifications were implemented by reusing the OpenEMed components. HL7 was more or less automatically implemented because the OpenEMed components use HL7 objects where relevant. Furthermore, standard components like OpenORB and the Resin web server and standard technology like SSL were implemented.

Of the OMG HDTF specifications, the PIDS is the most elaborated, i.e. it provides most functionality and probably covers every imaginable use case in the domain. This is partly based on the domain itself: identifying persons requires a small and certainly finite number of data and a limited number of different actions on that data. PIDS is generally acknowledged as the standard on patient identification.

The COAS specification is a read-only interface to the underlying medical data source. Although it provides a very flexible structure to wrap the medical information (the ObsData structure), it provides no specification on how to wrap the information. This can result in incompatible structures holding the same information (e.g. a blood pressure is structured in different ways). Archetypes can partly remove this problem by providing a more precise definition of the medical concept.

5.2 ARCHETYPES

The simplest possible implementation of archetypes focuses on the database-independent definition of medical concepts. It does not take into account possible links between archetypes and between the various instances and versions of an archetype.

We have not yet found any reason to add complexity to this implementation. This is true for the definition of the data as it is stored in and retrieved from the COAS server.

Using archetypes does not remove the possibility of defining different archetypes for the same concept, but this problem can be solved more easily:

- Archetypes are defined, not by the single user, but by a large representing group, e.g. the national association of general practitioners. Consensus on the definition prevents a myriad of variations.
- Archetypes are defined by medical experts separate from the software, while COAS has a strong link to the software generating or maintaining the data, which will often result in software engineers defining the COAS structure based on the structure of the underlying data. Therefore, using archetypes, which will more readily be accepted by the medical users, can also provide a layer that hides proprietary data structures.

Other problems to note in the use of achetypes are versioning and data exchange between systems. The versioning problem occurs when the definition of an archetype changes, e.g. in the number of fields. Batch conversion of data from one version to the next is rarely possible, which results in implementing functionality to handle multiple versions. Since domain knowledge is extracted from the software and moved into the archetype definition, ideally version interpretation should be added to the archetype definition too. This however, significantly increases the complexity of the archetype definition.

Data exchange between systems gives rise to problems in the integration of similar data created with different archetypes (e.g. how to compare blood pressure data created with two different archetypes).

Although the archetype concept is not yet qualified as a standard, it has all the potential to become one. The CEN/TC251 Workgroup 1 is working closely with the *open*EHR team to integrate the archetype concept in 13606. Both CEN/TC251 and HL7 have their own version of these data structures which are called General-Purpose Information Components (GPICs) [14] and Common Message Element Types (CMETs) [15], respectively. Currently there is collaboration between CEN and HL7 on GPICs and CMETs to define a harmonized set of data types.

As stated, our implementation of archetypes is very simplified. Yet, it is powerful enough to prove its flexibility. Further flexibility should be proven by the ease of the change of domain from stroke to hematology.

5.3 REUSE OF (OPEN SOURCE) COMPONENTS

Reuse of open source components greatly reduced the time necessary to develop our system. They also helped avoiding reinventing the wheel, one of our goals.

We even contributed some improvements and extensions to the OpenEMed project.

The software in its current setup has shown its robustness during development and usage since there has never been any point in time where the PIDS servers or the COAS servers crashed due to software errors. In fact, the only reboots were performed due to external conditions.

The flexibility of the software was demonstrated by the fact that we have built an entirely different EHR based on the same components as the original authors of OpenEMed with only minor extensions that were generic enough to be incorporated into the original code. Furthermore, the original authors have already built several EHR systems for several domains based on the same components.

Open source is often provided as the ultimate solution for vendor lock-in, security risks and lack of quality. There is not a single company with its own priorities; the source code is available for study of potential risks and generally, the open source

community responds more quickly to problems and feature requests. In general, this is true, but it only holds for the larger communities. When the community is small and/or the project is not financially supported, the priorities and the development speed of the community are much less flexible.

5.4 FIRST EXPERIENCES IN PRACTICE

The implemented version was evaluated by a team of four users, a physiotherapist, a speech therapist and two coordinators. Evaluation was done in an informal lab setting where all four where equipped with a laptop with a GPRS Internet connection. After a short introduction (less than half an hour) on the use of the laptop, the software to manage the GPRS connection and the system, they were all able to log in to the system, enter data, and review each others data. This session lasted approximately 3 h in total.

Some observations were made:

- The system performed well in handling multi-user access. There were no crashes due to invalid data entry and all data entered by a user was correctly retrievable by the others.
- The users were quickly able to find the relevant information, even the least experienced person.
- Sometimes the performance of the system was very low. We were not able to reproduce that using another PC with a fixed (i.e. not mobile) broadband connection simultaneously. Further investigation of the quality of the GPRS connection is ongoing.
- Userfriendliness had sometimes suffered from the attempt to be as precise as possible. The most prominent example is the way the users had to enter a date: the system was built to accept only a full date (consisting of day, month and year) in a fixed format, while the users preferred to use various formats, and/or incomplete dates and have the system convert it to the correct full date.

These observations show that the system is stable enough for everyday use and the concept of simulating the existing paper forms provides a smooth transition for inexperienced users to get familiar with the system. It is common for users to change or further specify requirements once they get to use the actual system, so comments on the system were anticipated. Overall they liked the system; the date example mentioned above was merely due to a misunderstanding from both parties: the developers forgot to take practicality into account by insisting on very precise date stamps, while the users did not realize that web applications using HTML in standard browsers cannot implement all user friendly features of a stand-alone application. The form definition design proved not to be flexible enough.

While the users started to use the laptops in a live situation, we decided to start on a next version of the software. This version should remove the problem of the form definition, add some extra user-friendly features and enhance performance.
One form definition integrated the query definition for the servers and the screen representation of a form for both retrieval and data entry. The latter proved to be too restricting. It was not possible to define different layouts for retrieval and data entry without increasing the complexity of various parts of the software.
The Apache Cocoon framework provides an approach that addresses these issues.

> *"Apache Cocoon is a web development framework built around the concepts of separation of concerns (making sure people can interact and collaborate on a project, without stepping on each others toes) and component-based web development." (http://cocoon.apache.org/).*

It uses the notion of "pipelines" to transform XML documents into various output formats, ranging from HTML to PDF to Microsoft Excel spreadsheets. Cocoon's flexibility shows in the possibilities to handle any kind of XML-ized source (e.g. results of a query to a database) and the ability to combine pipelines.
Part of Cocoon is a flexible framework for defining input forms that can solve the problems with the inflexible form definition and the user friendliness issues described above. Cocoon's separation of concerns provided a framework to further separate the various kernel parts as described in § 4.2.2.
After evaluating the version based on Cocoon, we will start on implementing the second domain (hematology). For this domain the DSS connection will be implemented and the PIDS server needs an interface to the hospital information system.
RAD and possibly a more functional implementation of LQS will be implemented subsequently.

6 Acknowledgements

The PropeR-project is funded by a stimulation program for ICT in healthcare (ICZ) of the Dutch Organisation of Science (NWO). We like to express our gratitude to the Vodafone Netherlands Foundation for providing the GPRS hardware and costs. And most of all we like to thank the OpenEMed team for the steady support and quick responses during the long period of working with the code.

7 References

1 Tange H, van der Linden H, Sas P, Beusmans G, Talmon J, van Oosterhout E, et al. Towards a PropeR combination of patient records and protocols. Int J Med Inform. 2003;70:141-8.

2 van der Linden H, Boers G, Tange H, Talmon J, Hasman A. PropeR: a multidisciplinary EPR system. Int J Med Inform. 2003;70:149-60.

3 Object Management Group. [cited December 2002]; Available from: http://www.omg.org

4 Lexicon Query Service Specification. 2000 [cited December 2002]; Version 1.0: Available from: http://cgi.omg.org/docs/formal/00-06-31.pdf

5 Person Identification Service (PIDS) Specification. 2001 [cited December 2002]; Version 1.1: Available from: http://cgi.omg.org/docs/formal/01-04-04.pdf

6 Resource Access Decision Facility Specification. 2001 [cited December 2002]; Version 1.0: Available from: http://cgi.omg.org/docs/formal/01-04-01.pdf

7 Clinical Observations Access Service Specification. 2001 [cited December 2002]; Version 1.0: Available from: http://cgi.omg.org/docs/formal/01-04-06.pdf

8 HL7 Reference Information Model. [cited 14 January 2003]; Version 1.19: Available from: http://www.hl7.org/Library/data-model/RIM/modelpage_non.htm

9 Health Level Seven Implementation Support Guide for HL7 Standard Version 2.3. 1998 [cited 2003 January 2003]; Available from: http://www.hl7.org/Special/IG/final.pdf

10 OpenEHR. [cited 4 November 2002]; Available from: http://www.openehr.org

11 Good European Health Record London. [cited December 2002]; Available from: http://www.chime.ucl.ac.uk/work-areas/ehrs/GEHR/

12 OpenEMed. [cited December 2002]; Available from: http://www.openemed.org

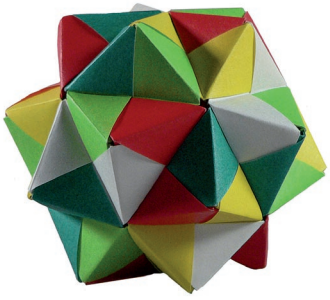
13 WHO. Internationale classificatie van het menselijk functioneren. Bilthoven: Bohn Stafleu Van Loghum; 2002.

14 Health informatics – General purpose information components – Part 1: Overview. 2002 [cited 2004/04/28];
Available from: http://www.centc251.org/Witems/PT/41/PT41-GPIC-Part-1-Overview-Draft-prEN.pdf

15 HL7 page. 2002 [cited 2004/04/28]; Available from: http://www.openehr.org/standards_hl7.htm

Chapter 5

Evaluation



Evaluation

1 Introduction

The previous chapters described the general architecture and the implementation considerations for PropeRWeb. PropeRWeb was designed to be a generic system that can be configured to meet the requirements of a specific Electronic Health Record (EHR) for a particular group of patients. In the PropeR project two application domains were planned for: cerebrovascular accident (CVA) and Leukemia. The CVA application has been used to assess the main principles of PropeRWeb. The Leukemia domain was used to address the issues around decision support and the link between EHR and the decision support system, in both technical and semantic sense.

The scope of this chapter is to describe the two implementation versions of PropeRWeb that were developed over time and their evaluation by users from the CVA domain.

First, the clinical context is described in which PropeRWeb-CVA was expected to operate. Next, the two different implementations will be described in more detail. The following section provides a description of the three different evaluations that have been made of the main principles of PropeRWeb. This chapter ends with a discussion of the findings and the lessons that could be learned from this study.

2 Context of PropeRWeb-CVA

When the PropeR project started, the intention was to collaborate with a more clinically oriented CVA project [1] that ran in parallel. The scope of the clinical CVA project was to investigate whether specific CVA patients could be treated in their home environment directly after their discharge from the hospital, rather than in a rehabilitation care facility. The role of PropeRWeb-CVA was to support the caregivers in this project. Several disciplines are involved in this kind of home care: speech therapists, physical therapists, a CVA nurse and a care coordinator to name a few. PropeRWeb-CVA was intended to support the communication among these various caregivers, primarily by providing access to a shared EHR. It was also expected to be available both within and outside the hospital. In the hospital, the patient data would be entered in the system prior to discharge. The home care providers would be able to view this clinical information and enter their own new data and observations in the record as soon as they started to treat the patient at home.

Prior to the project, it was estimated that yearly approximately 300 patients in the catchment area of our academic hospital, the Maastricht Academic Hospital (azM), would suffer from a CVA, of which 200 would be admitted to the hospital. Half of these 200 hospitalized patients were expected to receive home care. To be included in the clinical CVA project, at least two different disciplines had to be involved in the treatment of the patient. With this setting in mind, work was started along two lines: a further analysis of the situation and the implementation of the PropeRWeb design.

As described in Tange et al. [2] the existing situation was investigated using a combination of methods:

- First, daily practice was observed, which varied from observing the practice of nurses to attending meetings where patients were discussed.
- Second, the paper-based shared patient record for stroke rehabilitation that was introduced in the clinical CVA project was examined and described in detail.
- Third, the results from the first two activities were discussed and validated in open-ended interviews with some key persons. In the home-care setting, these persons were the team coordinator, a physical therapist, a speech therapist, an occupational therapist, and a community nurse.

The care process was described as a flowchart. The desired functionality of the EHR was described in UML use cases.

Based on these use cases and additional discussions with a care coordinator the decision was made to implement in PropeRWeb-CVA a set of forms of the shared paper-based record. This paper-based record was kept and updated at the patient's home.

The paper-based record was only partially used in the clinical CVA project. Reasons for not using were double recording (in both the paper record and the care provider's own system) and poor availability (the shared record had to stay with the patient). [2] The intention of the PropeRWeb-CVA system was to eliminate as many drawbacks of the paper-based record as possible, such as the availability issue and preferably the double recording. Timeliness of information, the availability of the information to the team members would also improve. More specifically: discharge information would be available to the team members before the first encounter with the patient.

The Dutch Heart Foundation has issued guidelines to be used when treating CVA patients [3]. The multidisciplinary team decided to implement these guidelines

gradually in the process, starting with a subset that focused on communication [2]. It was not possible to convert the guidelines to a computational version due to the nature of the guidelines. In general, the guidelines described that other disciplines were to be informed in a timely manner. This could not be implemented in a decision support component per se, but was the general function of the PropeRWeb system. This removed the necessity of implementing a decision support component for the CVA application in the first implementation.

3 PropeRWeb implementations

As described in chapters 3 and 4, the PropeRWeb software is based on open source components, implementing standards. The OpenEMed project [4] provided open source implementations of the OMG CorbaMed specifications such as the Patient Identification Service (PIDS) [5] and the Clinical Observation Access Service (COAS) [6], which are discussed more elaborately in chapter 3. The PropeRWeb software is web-based to allow easy access from multiple locations. The multidisciplinary team would be provided with a laptop with Internet access to allow them access to the patient's record while visiting the patient, but also in other locations and time frames.
The system had to be used as a shared EHR record, unfortunately without integration with the various systems already in use by the team members, since none of these systems was open enough to make a connection.

3.1 ARCHITECTURE

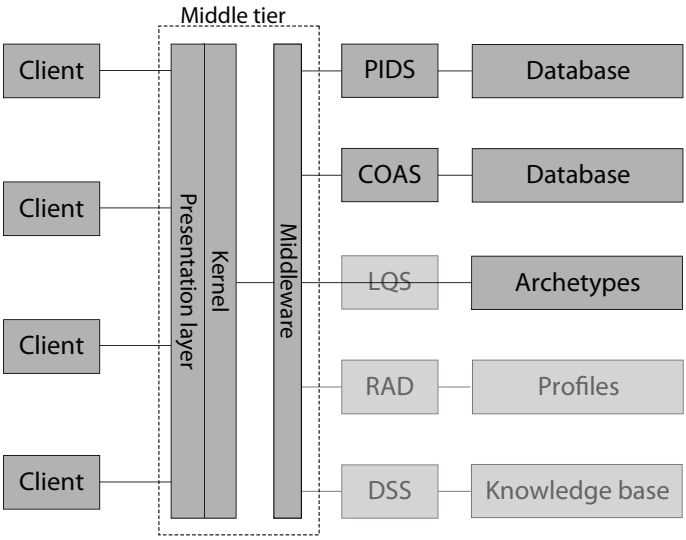
Figure 9 shows the architecture of PropeRWeb. In PropeRWeb-CVA , several components were not implemented. As stated before, the DSS component was not necessary. The OpenEMed implementation of the Resource Access Decision (RAD) was under development and not stable enough for use, which led to the decision to postpone the implementation in PropeRWeb. The implementation of the Lexicon Query Service (LQS) component was also not stable enough for use. The functionality of retrieving archetypes was therefore implemented in the middle tier as discussed below.

3.2 COMPONENTS

3.2.1 PIDS

The PIDS implementation of OpenEMed was used, which provided a nearly complete implementation of the OMG PIDS specification. The available implementa-

FIGURE 9 Architecture of PropeRWeb, PIDS = Patient Identification Service, COAS = Clinical Observation Access Service, LQS = Lexicon Query Service, RAD = Resource Access Decision, DSS = Decision Support Service.



tion was found to be sufficient to meet the requirements of the CVA clinical service and of PropeRWeb.

The PIDS component is launched as a CORBA service. The OpenEMed components provide connections to several databases, including HSQLDB [7]. The HSQLDB database has a small footprint and is available as open source, which was the reason for choosing it in PropeRWeb.

Two instances of the PIDS service are used by PropeRWeb; one for the patient information and one for the health care provider information.

A PIDS profile consists of traits. As stated in chapter 4 traits can be a simple key-value pair or a composition of several traits. The notion of care team was implemented using the PatientRelation trait. Although the OpenEMed implementation supported the use of composite traits (i.e. traits consisting of traits), additional code was needed to implement search and manipulation functionality of these composite traits. This code, developed by the author, was subsequently accepted and adopted by the OpenEMed community.

The PatientRelation trait defines the relation between the health care provider and the patient as well as the start and end date of the relation. The Role subtrait was used to describe the profession of the provider and therefore the PatientRelation trait was added to the provider PIDS, not the patient PIDS.

The code to support composite traits was the only major change in the OpenEMed code. It was sufficient to modify settings in the configuration file to adjust PIDS to the PropeRWeb environment.

3.2.2 COAS

The COAS implementation of OpenEMed was also used. The original OMG specification of COAS provided only read access. The OpenEMed team had extended the implementation by adding write access as well. This implementation provided enough functionality to meet this project's requirements.
The COAS component was also launched as a CORBA service, also using an HSQLDB database. Like the PIDS component, it was sufficient to modify the COAS configuration file to adjust COAS to the project environment.

3.2.3 Archetypes and forms

Based on user interviews the decision was made to implement the existing paper forms with only minor variations. This resulted in a collection of thirteen forms, each consisting of independent elements. The elements were implemented as archetypes.
As described in more detail in the chapters 3 and 4, the archetype approach is based on the idea that the underlying database implements a generic data model (reference model). The archetypes are used to specify the clinical concepts by constraining this data model. This is commonly referred to as a dual model approach. [8]
The data model was based on the COAS specifications. COAS specifies observations (ObsData) as entities (see chapter 4). An ObsData structure consists of qualifiers (ObsQualifier), holding metadata such as authors and timestamps, ObsValues, defining key-value pairs, and/or ObsData structures.
The archetypes were mapped onto COAS observations. Forms were used to group instances of archetypes at a form level for data entry, editing and overview. The forms were also mapped onto COAS observations.

Our implementation of archetypes tried to follow the specifications that were available at the time, as closely as possible. At the time, only a rudimentary kernel, written in Eiffel, was available that could interpret the notation in which the archetypes were written. This notation was later extended to become the Archetype Definition Language [9]. Since the kernel did not provide all the functionality necessary for the PropeRWeb system and since the majority of the archetypes necessary for the CVA domain were not available and had to be defined in this project, the decision was made to write custom archetype definitions in XML and write a custom parser.
Development was started with the simplest possible implementation with the intention to increase complexity when necessary.
Each archetype was defined by an XML structure with a name and a data type.

When relevant a format was added to a data type, in particular to represent dates. For each form, a separate XML file was created that defined the elements (i.e. archetypes) and the display information. The display information would define the display label and the input format as well as additional help information. A form could have multiple instances, similar to multiple sheets of a paper form. An instance is created when the user selects 'new form'.

Each form definition shared the same set of metadata:

- Start and end timestamp of the observation
- Timestamp of the data entry
- Author of the observation (i.e. the person responsible)
- Creator of the data (i.e. the person performing the data entry)

The choice of metadata was based on OpenEMed examples that were used in real-time scenarios as well as the assumption that the observation could be done at a different time by a different person than the person entering the data on the observation in the system. Although the timestamp of the observation was stored at element (i.e. archetype) level, it was assumed that the observations of a set, combined in a form, were obtained at the same time, therefore allowing the users to enter the timestamp of the observation once, at form level.

3.2.4 Middle tier version 1

The middle tier, as shown in Figure 9, consists of three components: the middleware, the kernel and the presentation layer. The middleware component consists mainly of the standard CORBA services, such as the naming service and the trader (the central services to locate connected services). The OpenORB open source implementation was used.

The kernel was implemented in Java, using plain Java classes. The kernel provided the connection with the PIDS and COAS services and managed the conversion of the queries and the results. The results were passed to the presentation layer as a DOM[7] tree.

The presentation layer was implemented in Java Server pages (JSP). It converted the query results to HTML by applying an XSL style sheet and adding the more static page elements such as the demographic data of the selected patient and the menu in the sidebar.

The form definition files specified not only the archetypes that are used by the

7 The Document Object Model (DOM) is a generic, hierarchical, object oriented representation of an XML structure.

form, but defined also the display and data entry presentation of each included archetype.

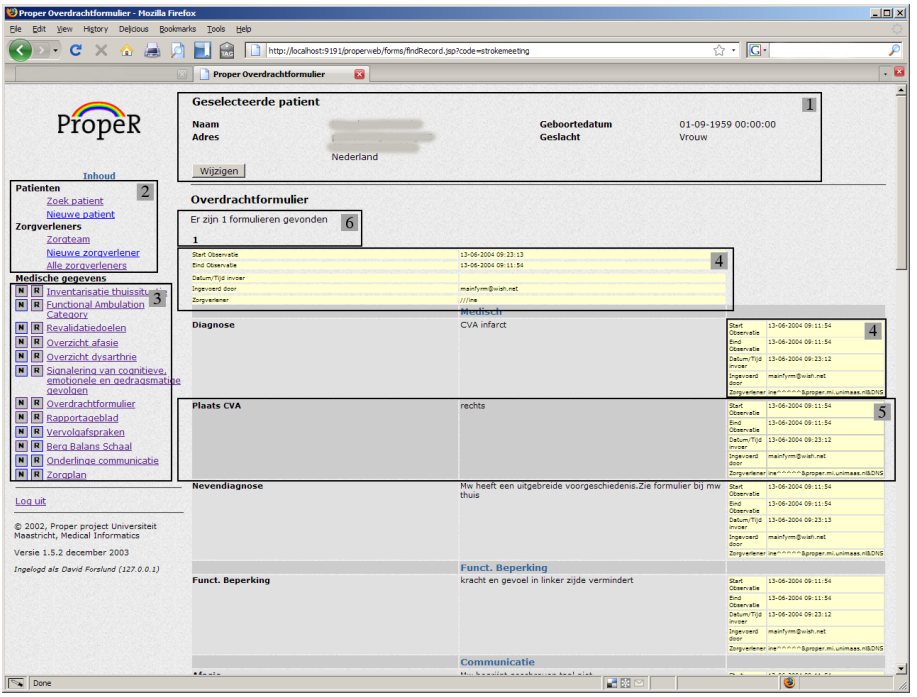
After starting the system, the kernel reads all the definition files of the archetypes and of the forms. In memory, object trees are built of a generic archetype and of a generic form. The form object tree is modified to reflect a specific form and populated with the archetype object trees, copied and modified based on the archetypes specified in the form definition.
Depending on the request (i.e. data entry or data retrieval), the archetype objects are left empty or are populated with the information retrieved from the database by the COAS component.
To accommodate users, when a new form instance was requested of a form with existing instances, the most recent instance was retrieved to pre-populate the new instance. In all cases the metadata of the form and the archetype objects are preset to reasonable values such as the current date and time for both the observation and the data entry and the name of the logged in user as both author and creator.
A data storage request results in a pruning of the form object tree, to remove empty archetype objects and a subsequent extraction and conversion of the information to a COAS structure, which is subsequently stored in the database. The object tree of the form itself is also stored but contains only references to the archetype objects.
Multiple instances of a form are allowed, analogous to the existence of multiple copies of the same paper form. The most recent instance hides all prior instances. This mechanism supports an audit trail that prohibits deletion of information.

After logging in, the user would search for a patient, by entering demographic information in a search window. The system queries the patient PIDS server and displays a list of patients that satisfy the query. After selecting a patient, the most recent value of the stroke meeting form was retrieved and displayed. The stroke meeting form definition contains a set of elements (i.e. archetypes) that describe discharge information of the patient for the multidisciplinary team. Users could then select the appropriate form from the list in the sidebar. For each form, they could select a list of all available form instances, a new instance of the form for data entry, or the most recent instance of the form.
Figure 10 shows an instance of the stroke meeting form in the first version of PropeRWeb.

3.2.5 Middle tier version 2

During the test phase of the first version, the software was redesigned to address

FIGURE 10 Screenshot of the first version of PropeRWeb, showing the stroke meeting form, 1=Patient information from the Patients PIDS server, 2=functions on the PIDS servers, 3=forms, stored in the COAS server, 4=metadata at form level and at archetype level, 5=archetype, 6=available form instances.



some usability issues. These consisted mainly of a performance problem and some issues in the presentation layer, which are discussed in more detail in § 7.3.

In this version, the kernel and the presentation layer were rewritten using the Apache Cocoon Framework [10]. The use of Cocoon was expected to solve these usability issues. Cocoon provided much of the required front-end functionality that had required coding in the first version out of the box. Since Cocoon was developed specifically to be a framework for web applications, it already contained functionality to easily configure and manipulate input forms and to query, retrieve and publish information from various data sources. The available functionality was used as much as possible. Cocoon did not support PIDS or COAS services; therefore, the appropriate classes from the first version were adjusted to the requirements of Cocoon.

This version was not only adjusted to use Cocoon, but also introduced some additional changes to be more compliant to the objective of domain independence, as described below.

3.2.5.1 *Increased domain independence*

The Java code was rewritten to be more generic and less domain dependent. For example: version 1 had classes defining patient, gender and patient relation. In the second version, they merely followed the PIDS definitions of person and traits. This version also created a more prominent separation between the code that handled the connection with the PIDS and COAS services and the code that handled the internal representation of the forms and the persons.

3.2.5.2 *Separation of domain model and presentation layer*

A more clear separation was made between domain model definition and presentation layer definition. The archetype definitions were moved into a separate file, representing the domain model. The archetype definition was extended with possible values. The form definition file was reduced to a list of references of the included archetypes and served as a template for instances (which in principle would permit any given archetype to be re-used across multiple templates and therefore forms).
In the presentation layer, as much functionality as possible from the Cocoon framework was used. The data entry format was separated from the display format of the forms. The data entry format was built using the CForms [11] configuration files, while the display definition used the JXTemplate [12] configuration format. CForms is introduced later in this chapter in § 7.3.1. JXTemplate is a generator that allows the use of Java and JavaScript objects to be used in a web page. Both formats simplified the definition of the layout of the forms.

3.2.5.3 *Revisions*

This version also introduced the notion of revisions. In the first version, a form could have multiple instances, but it was unclear if these instances were in fact the result of multiple data entry and save operations of a single data set (i.e. a single sheet of the corresponding paper form, which is edited several times) or of multiple data sets (i.e. multiple paper sheets). This was corrected in the second version by adding the edit functionality to forms and the creation of versions. If the user selected *new* to create a new form, a new instance of the selected form was created. Selecting *edit* would create a new version of the selected instance.
While the first version of PropeRWeb would list all available instances of a particular form, the second version only showed the most recent version of all available instances, therefore correctly following the concept of the paper forms. The revision was determined at the archetype level, i.e. a new version of a form would be created if at least one of its elements was modified. Only the modified element would have a new version, the version of the other elements would remain the same.

TABLE 2 Implemented forms, '# visits' refers to the number of times the forms are retrieved in the PropeRWeb application

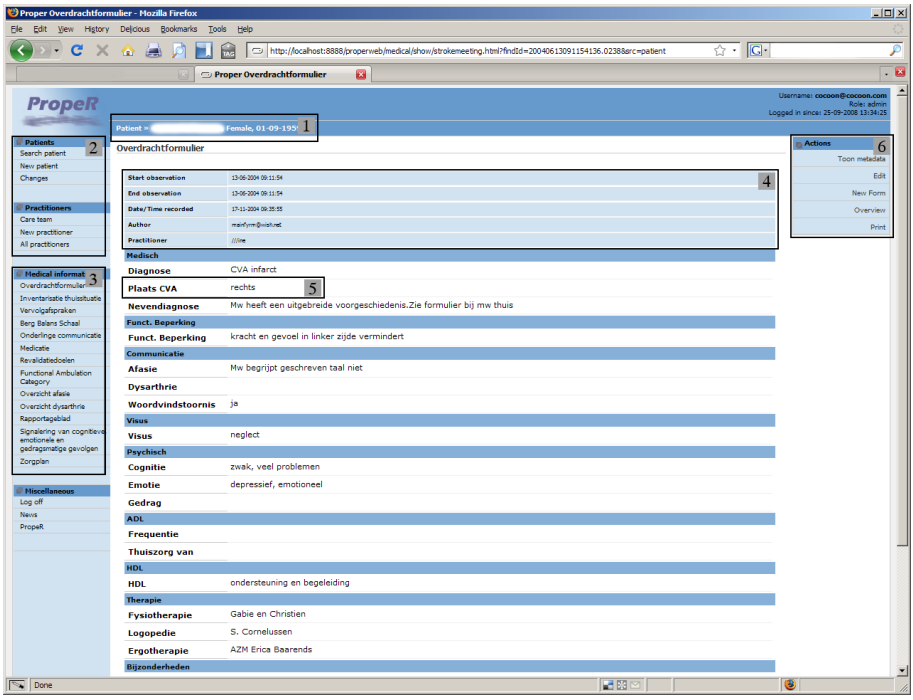
Form	Description	Discipline	# items	# visits
Stroke meeting	Discharge information	General	21	60
Home situation	Information on the home situation of the patient	General	25	19
Care goals	Goals for the various disciplines	General	8	23
Signaling	Test form for signaling various problems	General	26	3
Reports	Information by various care providers, not necessarily a team member	General	1	15
Appointments	Appointments of the patient with various care providers, not necessarily team members	General	1	1
Communication	Questions and answers to/from other team members	General	3	21
Care plan	Frequency and care of other, non team members	General	6	3
Medication	Medication prescription and administration information	General	1	3
Aphasia	Test form for aphasia	Speech therapist	46	0
Dysarthria	Test form for dysarthria	Speech therapist	58	6
Functional Ambulation Categories (FAC)	Test form for FAC tests	Physiotherapist	14	11
Berg Balance Scale	Test form for the Berg Balance Scale	Physiotherapist	16	2

3.2.5.4 *Additional functionality*

Based on user feedback one form was divided into two forms (see Table 2).
Finally, the layout of the web pages was changed to optimize the use of screen real estate. A context-related menu was visible on the right hand side, providing the available functions, such as new, edit, view and print. The latter would create a view optimized for printing, with the intention to avoid manual copying of the data in the paper-based record.

The use of the system remained the same, although the available functions were selectable in the right sidebar. An extra function to display changes since the user's last login was also available. Figure 11 shows a screenshot of the second version of PropeRWeb. The form shown here is the updated version of the one shown in Figure 10.
Appendix B shows an example of the various definition files. Version 2 of PropeRWeb was deployed towards the end of the trial.

FIGURE 11 Screenshot of the second version of PropeRWeb showing the same strokemeeting form, 1=Patient information from the Patients PIDS server, 2=functions on the PIDS servers, 3=forms, stored in the COAS server, 4=metadata at form level (archetype level is hidden), 5=archetype, 6=context menu. Form instances are selected in an overview screen, prior to this screen.



4 Evaluation methods

Three evaluations were performed:

1 Analysis of the use pattern of the PropeRWeb-CVA system,

2 Qualitative analysis of the usability of the PropeRWeb-CVA system,

3 Technical assessment of the system

- From a developers perspective
  - Use of services
  - Separation of concerns
  - Archetypes
- From a reusability point of view
  - Various implementations
  - Archetypes
  - Tools

Analysis of the use pattern should give an indication of the usefulness of the system to the users. This analysis was conducted by reviewing the information entered in the system by test users. It was not possible to evaluate the completeness of the data entered, because there was no reference on paper or otherwise available to the researcher. Therefore, only the forms with instances were counted.

The second analysis focused on how useful the test users themselves found the system. For this analysis, each participant was interviewed. The interviewer was not the developer, to allow the interviewees to speak freely. Users were asked which version they used and their experiences with the system. Appendix A shows the mind map that was used to guide the interview. The interviews were transcribed and subsequently analyzed with a focus on the user's appreciation of the usability of the software, the availability of the information (i.e. the benefits of having the patient information available) and the perceived differences between both versions of the software.

The third evaluation focused on the technical assessment of the software, from a developer's perspective and from a reusability point of view. The intention was to analyze the technical choices made and whether it was possible to create an application for a different domain and evaluate the amount of effort necessary.

Three different persons, with different levels of computer skills and different levels of knowledge of the software, were asked to configure the second version of the system for a different domain.

TABLE 3 Forms entered

U1 and U4 are care coordinators, U3 is speech therapist, U2 is physiotherapist (between brackets are the number of forms entered).

PatientID	No of forms entered	Users entering data
122	0	
62	1	U4
104	2	U1
105	3	U1
142	1	U1
103	5	U1 (1), U2 (4)
23	3	U1 (2), U4 (1)
82	2	U1
2	2	U1
102	1	U1

5 Use pattern analysis

The first version of the system was tested by four test users who were provided with a laptop, equipped with a GPRS card, graciously donated by the Vodafone Netherlands Foundation. The four test users were two care coordinators, a speech therapist and a physiotherapist.
They received a training of two hours. During the training, they could familiarize themselves with the laptop, the use of the GPRS connection and the application. The developer was available for explanations and questions. The software was deployed on a server located at the Medical Informatics department of the Maastricht University.
The second version was tested by one care coordinator.

5.1 RESULTS

Twelve forms[8] were implemented in the system for the CVA application. Of these forms, two were specifically intended for the physiotherapist, two were specifically intended for the speech therapist. The remaining eight were general forms for registering information necessary for the entire team. Table 2 shows the list of forms with a description and the relation to the discipline.
Ten patients were included in the clinical CVA project in a period of 10 months, and the evaluation therefore focused on the user experience of PropeRWeb for these ten patients (the reasons for this low number of patients are discussed in § 5.2). Three out of four users entered data. The speech therapist did not use the system, because she was not part of the care team of the included patients. Table 3 shows the results. On three occasions a form was entered multiple times, but counted as one instance, for the following reasons:

- The information was identical;
- The timestamps of the data entry were not more than a few minutes apart;

The first version of the software (used to create this information) did not support an edit function, while the new function created a new form instance preset to the previously entered data of the most recent form instance.
We therefore assume that the users either accidentally or for safety created multiple instances of the same information set.
The main user was one of the care coordinators (U1). She had entered almost all information. Information was entered by the physiotherapist (U2) for only two patients.
Table 2 shows the number of visits of the various forms. The forms used most are

8 In version 1, Careplan and Medication were combined in one form.

the forms that focus on communication between the team members and the discharge form. The low overall numbers can be explained by the fact that there were no frequent visits to the patients, and by a pattern of looking up information around the time of visiting the patient.
In PropeRWeb-CVA version 2, functionality was added to show users what information was added or changed since their last session in the system.

5.2 DISCUSSION

Although original estimates were for eight to ten patients per month, on average only one patient per month qualified for inclusion in the PropeR CVA project. The main reasons for the low number were:

- As stated before the PropeR project worked in partnership with a more clinically oriented CVA project. The intention was to support the care of the patients included in the clinical CVA project with the PropeRWeb system. However, since patients were only included in the CVA project when they needed help from at least two therapists from different disciplines, a large percentage of the CVA patients was not included in the CVA project and therefore not available for the PropeR project.
- In the initial trial only four laptops were available for use. Two care coordinators, a physiotherapist and a speech therapist were included in the trial. This resulted in the inclusion of only patients who were included by the CVA project *and* who received treatment of at least two of the users. Unfortunately none of the patients included were treated by the speech therapist during the trial period, which explains why the speech therapist (U3) has not used the system.
- At the start of the project, patients were discharged to their home environment. This situation changed just after the user trial started. The new strategy was to discharge patients to a triage department in a nursing clinic. From the triage department fewer people, who needed less care, were discharged to their home environment. The rest moved to different departments in the nursing clinic.

6 Qualitative analysis of the usability of the system

6.1 OVERALL ASSESSMENT

The PropeRWeb software was considered useful and allowed easy navigation to the correct information. The users agreed that the forms covered all the relevant information about the patient. One person even remarked that it was too much

information, even though the system was a copy of the paper forms used.
All users informed us of a new system that was to be implemented in the near future. The system would not have any medical information, but merely served as a patient tracking system. The interviewed users all stressed that the PropeRWeb software was more useful to them.
Although the neurology department of the hospital initially agreed to provide the discharge information, they were no longer able to do so when the trial started, due to changes of the key persons as well as changes in the hospital's security and privacy policies. They could only provide discharge information on paper to the primary care coordinator. This effectively reduced the usefulness of the system, because all discharge information had to be entered by hand and became available only after the actual discharge of the patient, rather than slightly before.
However, not the timing but the effort needed to manually enter the information was considered a problem. This issue was further aggravated by the inability to connect the PropeRWeb system to the respective systems of the various users and various special purpose registry systems, thus forcing them to enter information in two and sometimes three different systems. During the design phase, attempts were made to cooperate with the vendors of the various systems already in use. These attempts failed due to the very defensive attitude of some vendors (one vendor flatly refused) and the fact that at least one system had no import or export functionality at all. Even cooperation from the vendor would take too much time to implement the functionality.
During the test period, yet another registration system, for billing purposes, was introduced that the care coordinator was required to use. The print feature of PropeRWeb version 2 managed to avoid manual update of the paper record still in use at the patient's home.

6.2 TECHNICAL ISSUES

During development, the software was tested frequently on a wired network connection with good results. The GPRS specification showed a high enough speed for the intended use, which was confirmed with a test setup in various locations. GPRS coverage of the support area of the users was substantiated by a coverage map, showing 100% coverage in the Maastricht region.
Unfortunately, in daily practice, the speed often dropped and the users reported no GPRS coverage in certain parts of the region. The combination of a slow and unstable GPRS connection and a subsequent suboptimal rendering of the required page, confronted the users often with a timeout, which led to not being able to access the information or, in case of data entry, to loss of data. A work-around was provided by informing the users how to access the PropeRWeb system from their

own computers with broadband Internet connections. This solved most of the performance issues.
The GPRS problem and the absent connection to their own systems made them revert to using email as their primary digital communication method.
This, and the long development time, led to the fact that only one user (U1) has used the second version of the software. This version was perceived to be more user friendly and solved many of the previously encountered problems. However, the major difference in layout and colors (a clear distinction between the versions) could not be recalled, which is probably due to a limited number of sessions with the latest version.

6.3 DISCUSSION

Although the system was perceived as useful and user friendly, the lack of a reliable wireless connection and the lack of integration with the users own systems proved to be an unanticipated hurdle. The high-speed wireless connections that are currently available would probably have stimulated the users to keep using the system during the trial.
Bigger problems were the organizational changes that led to fewer patients to include and the larger efforts that had to be undertaken to get information into the system.

7 Technical assessment

7.1 USE OF SERVICES

7.1.1 PIDS

The PIDS implementation of OpenEMed was not a complete implementation of the OMG PIDS specifications, but provided enough functionality for the PropeRWeb system. Before using the PIDS service with the HSQLDB database, we performed a scalability test by adding 1200 persons in a batch process to the PIDS server. Searches performed on these 1200 persons showed only a minor increase in time from the usual test set of 10 users. We anticipated that not more than 100 – 200 persons would be added to the servers during the trial period; hence, the software would scale well enough for the purpose of this application.
The use of the PIDS service required examination of the OMG specifications rather than a study of the actual source code. As an example: the name and address trait were defined as HL7 v2.3 strings with delimiters. This was implemented in the OpenEMed implementation. It took some time and additional study of the original

HL7 specifications to determine the specific order in which the substrings of each trait had to be stored.
The PIDS specification can be considered as showing the concept of a dual model approach: the demographic and identifying information of a person is stored in generic traits (i.e. generic building blocks), which are further specified to hold the name or the address of the person (i.e. specifications of demographic concepts). The HL7 v2.3 version of the demographic information is described in detail as an example of a specific implemention, and is also implemented in the OpenEMed code.
Although the specification mentions the version in the name of the module (HL7Version2_3), there is no specific attribute to show the version in the trait itself. This is inconsistent with the COAS specification of a Coded Element, which does not only specify which coding scheme is used, but also the version of the coding scheme.
The use of the PIDS service was straightforward and generic as it allowed for storage of patients as well as care providers without code modification. In fact, the specification is commonly recognized as one of the best specifications on person identifications and therefore often implemented in various platforms. This warrants the continued use of the PIDS component.

7.1.2 COAS

Like the PIDS specification, the COAS specification also builds on a generic recursive structure that can hold any information. The OMG COAS specification provides only read access, but the OpenEMed project has added edit and storage access. The built-in object-to-relational mapping handled the storage and retrieval from the underlying database. This overruled part of the built-in functionality of the database to handle multi-user access, which subsequently had to be added to the COAS implementation. For PropeRWeb the implementation was sufficient, but it would probably not scale well enough for a large population of users manipulating large amounts of data at the same time, due to the rudimentary implementation of multi-user access in COAS.
Contrary to the PIDS specification, the COAS specification is much less used and implemented. The advantage of the COAS specification is the generic structure that can hold any type of information and simplifies the exchange of information. However, the COAS specification is largely superseded by the format of other exchange formats such as archetypes and has therefore become superfluous.

7.1.3 CORBA

The implementation of CORBA in the OpenEMed services was functional enough to use without a deep level of understanding the inner workings. It proved to be a sta-

ble and fast connection. Adding security could be deferred to the CORBA Security Services [13]. In the mean time, the general trend was away from CORBA to SOAP [14] and web services [15], due to the complexity of CORBA and the emerging web based applications, even though no security functionality was available in the web services protocols at the time.
Although CORBA provides a more robust technology, its complexity and the general trend towards web services warrants a change of the PropeRWeb architecture to web services to enable broader uptake.

7.2 ARCHETYPES

Initially we decided to use the COAS ObsData structures to define the information. This posed two problems:

- Although the structure was generic enough to express all concepts, it was not possible to distinguish between types of structures; for example, there was no distinction between the concept of a form and a data element representing an item on that form.
- More importantly, due to the read-only focus of the COAS specification, there was no definition for the requirements of data entry such as the format of a date and a predefined set of values.

This led to the decision to adopt the concept of archetypes as proposed in the GEHR, Synapses and *open*EHR projects (see chapter 2).

7.2.1 Mapping

Mapping archetypes onto a COAS structure was straightforward, since they are both object-oriented structures. However, since the COAS structure is more generic, extra tags were needed to preserve the various types of structures available in the archetype definition. The separation between COAS structures and archetypes provided a separation between generic data storage and retrieval (COAS structures) and specific concept definitions (archetypes). In this context, the COAS structure could be considered as a very simple reference model to the archetypes.

7.2.2 Localization

Archetypes were emerging at the time of the PropeRWeb development. They did not provide any option for localization (i.e. conversion to different languages). This has been corrected later by adding numeric references to each element of an archetype and by adding a terminology section to describe the references in various languages and coding schemes.
In PropeRWeb, this lack of localization was solved by adding language specific descriptions to the presentation layer.

7.2.3 Modeling

As stated before, there were no CVA related archetypes available, so custom archetypes were modeled. As there was little evidence at the time on how to model archetypes, frequent discussions were held on the smallest unit of information. Since there was not sufficient modeling knowledge available in the test users, simple key-value pairs of unrelated items were defined initially. In hindsight some of the items that were modeled as separate archetypes, could have been grouped together, such as type and location of the CVA.

The same evolution in grouping and modeling can be seen in the archetype modeling community. A trend is visible towards grouping medical concepts that are often used together such as height and weight, even thought they represent distinct and unrelated concepts.

7.2.4 Reusability

Archetypes promote reusability by structuring an independent medical concept. The PropeRWeb application demonstrated this by reusing archetypes both within a single application and across applications.

7.2.5 Forms and templates

During the development of PropeRWeb, the need was felt to group archetypes into logical sets and to separate the presentation layer from the archetype definition. In hindsight, this implementation of forms was a very preliminary implementation of the templates of the *open*EHR community [16]. The latter groups archetypes into logical sets and presets various elements of an archetype to a default that is relevant for the anticipated context. The current view is to use templates, not archetypes as a base for the presentation layer, which conforms to the use of forms in the PropeRWeb system.

7.3 IMPLEMENTATION OF PROPERWEB

7.3.1 User-friendliness

As described in § 3 the initial version of PropeRWeb was based on components implementing PIDS and COAS. The front end was based entirely on Java Server Pages (JSP). Initial testing by the test users revealed that the software did not provide the expected user friendliness that was anticipated by the users. As an example: users expected various short hand notations available for entering a date, similar to the desktop versions of their own EHR systems. PropeRWeb only accepted one specific format of a date.

The CForms module of the Apache Cocoon Framework, on which the second ver-

sion of PropeRWeb was based, provided flexible and user-friendly data entry widgets that would provide more of the requested flexibility.
An additional advantage of the use of the CForms module was that it enforced separate definition files for presentation and for the data model (archetypes). This contributed further to the realization of a domain agnostic/generic EHR system.

7.3.2 Performance

The performance of the first version was sometimes insufficient. This was mostly experienced when the application was used on the laptop with a GPRS connection. The most prominent reason was the slow connection, but another issue was the build time of the pages. The first version, using JSP, built every page from scratch every time it was accessed, using a Document Object Model (DOM) [17] for the content transformation. The DOM creates an in-memory tree of the XML structure. It provides functionality to manipulate the tree to a great extent. Its counterpart is the Simple API for XML (SAX) [18], which processes an XML structure in a serial manner. Although the DOM is more flexible it is known that a DOM transformation is slow, compared to a SAX transformation.

Cocoon had optimized page rendering by implementing caching and SAX transformations. Cocoon's caching mechanism provided a better performance out of the box than the first version of PropeRWeb. The second version of PropeRWeb could provide web pages with a minor delay even on a slow connection.

7.3.3 Authentication

Another issue that emerged during the trial of the first version of PropeRWeb was the authentication component. Due to the stateless nature of the HTTP protocol, sessions are introduced to track the state of the user between various web pages. Sessions also hold authentication information to determine whether the user is allowed access to the page. The JSP authentication component was a simple implementation of the authentication tracking mechanism, while Cocoon provided an elaborate, yet easy to configure authentication module with automatic session timeout.
This allowed the development of a much more stable and much better performing application.

Feedback during the trial period from the care coordinator who used it, showed that the second version of PropeRWeb solved the initial problems of the first version.

7.3.4 Code reuse

Although a large part of the first version was rewritten to use Cocoon, the majority of the changes concerned the presentation layer. The design of the kernel as a whole was not altered, although the internal components were more prominently separated. The code to connect to the PIDS and COAS components had the smallest code changes.

There were also drawbacks. The major topics being the steep learning curve of PIDS, COAS, the inner workings of Cocoon and the requirement to connect Cocoon to the existing back end of PIDS and COAS components. The latter required custom developed code to connect Cocoon to the components and handle the objects passed between Cocoon and the ORB services. Although Cocoon was quite flexible and the connection was successfully implemented, the steep learning curve required a considerable amount of time to finish the second version.

7.4 SEPARATION OF CONCERNS

The essence of Separation of Concerns (SoC) is the alignment of logical functionality with component boundaries to improve stability, flexibility and design of the system [19]. SoC provides an important base for our objective of developing a generic system. Using the experience of version 1 SoC was improved in version 2 in several ways:

- The connection between the kernel and the PIDS and COAS services was made more explicit.
- The references to (demographic) domain concepts such as 'gender' and 'name' were removed in favor of more generic terms such as 'trait' and 'person'.
- The form definition was separated into a domain related definition, a composition of archetypes, and a presentation related definition — the display and data entry configurations.
- Forms and archetype instances and versions were introduced to distinguish between new data sets and edited data sets, respectively.
- The GUI definition was further separated into data display and data entry. In hindsight, the definition should have been on both the form and the archetype level, rather than just on the form level, since the latter obstructs an easy reuse of an archetype in a different form.

7.5 EASE OF REUSE

The system was designed with reuse as one of the main requirements. The objective was to make the software agnostic to the domain and define a data model (i.e. archetypes) and forms in files that were interpreted, rather than hard-coded.

To develop a new application with the second version of the PropeRWeb system, a set of configuration files was needed. This set consisted of:

- A single XML file containing definitions of all archetypes (data model).
- A single XML file per form with references to all the relevant archetypes (compositions of archetypes).
- Four XML files per form, one for the display definition and three for the data entry configuration of the Cocoon CForms module (presentation layer).

Examples of these files can be found in Appendix B.

To test the reusability, three persons implemented a new domain in a separate setup of the software (version 2). These persons built an MRSA registration application, a proof of concept for a clinical trial on cardiology treatments and a patient record for AML patients at the hematology department in the hospital.
All users used an XML-editor (Oxygen XML as a plug-in for the Eclipse editor) to create the set of configuration files.
The various implementations will be discussed.

7.5.1 MRSA registration application

The intention of the MRSA registration application was to register the admission of foreign patients in several hospitals in the Maastricht region along with information on the presence or absence of MRSA. The objective of the trial implementation was to evaluate the ease of implementation of a different domain application.
The application consisted of four forms, each containing 6 to 10 data elements. The application was built by a person (P2) familiar with software development.
After about two hours of explanation of the software and the syntax of the configuration files by the developer, P2 was able to start to write the required archetypes and forms specific to the application. P2 needed two short follow-up meetings to solve and/or clarify issues and was able to finish the application in a day.

7.5.2 Clinical trial

The intention of the clinical trial application was two-fold. It was built as a proof of concept, to validate the completeness of PropeRWeb as an EHR system. It also helped the trial coordinator to elicit requirements for the registration system that was eventually used for the clinical trial.
The implementation consisted of a single form that contained the inclusion and exclusion criteria. For each criterion the user could answer yes, no or unknown.
A JavaScript file was added to the form definition files that contained the algorithm to decide on the inclusion or exclusion of the patient.
Since this application was developed by the developer herself (P1), no additional

explanation was necessary. The form was finished in 4 to 6 hours, including the JavaScript.
The clinical trial was designed as a multi-center trial. This required a more rigorous access control mechanism than available in the current PropeRWeb implementation. This made the PropeRWeb implementation unfit for use, but it provided the elicitation of the actual requirements regarding access control in the clinical trial.

7.5.3 Hematology application

The intention of the Hematology application was to provide an EHR system tailored towards the AML patients that were admitted to the hematology department in the Maastricht Academic Hospital (azM). The objective of the trial implementation was the evaluation of the connection of PropeRWeb to a decision support system. A study into the issues of such a connection was published in [20].
The data model was quite large. The application consisted of 25 forms with the number of data items ranging from 1 to 54, with a mean of seven.

The person coding this application (P3) was an average computer user, who had no experience in developing software. The developer spent about 2 hours explaining the software and the configuration files to P3, after which she was able to start writing her own files. P3 needed several follow-ups to solve and/or clarify issues.
P3 needed several days to develop the application over a period of several weeks. This was partly due to the larger data model, and partly due to frequent discussions with the future users of the application.
Unfortunately, the trial ended prematurely because the implementer changed jobs. All forms were implemented but the application was never tested nor was any connection established to a decision support system.

7.5.4 Findings on reuse

All tasks were finished successfully. The XML-editor only provided general XML syntax checking, but it was up to the implementer to correctly reference the names of the archetypes and forms in the various files. There would be a gain in speed if specific editing tools were available that could assist in creating the files as well as reduce typing errors. Such tools would also make it possible to have less knowledgeable persons build applications. This need has also been noticed in the archetype community and has lead to the development of several archetype editors [21, 22]. These editors allow clinical experts to develop archetypes, but focus on the data modeling aspect, not the presentation aspect. As said in § 7.2.5 the recent development of templates for archetypes [16] addresses the presentation aspect and has led to the development of a Template Designer [23].

In the current system, authentication was implemented using only a user/password login screen and an automatic session timeout. No further authorization was implemented yet. The MRSA application and the clinical trial application showed the need for a more fine-grained security implementation. Both applications were intended to be used in different clinical centers where each center was only allowed to review their own data. The clinical trial application required even more access roles within the respective centers and across the centers. It proved to be very difficult to add security as a separate component (RAD) to the existing software. Access restriction influences all layers of the software, from restricting access to data retrieved from the database to hiding unauthorized actions in the presentation layer.
The requirements and the underlying issues are discussed more thoroughly in chapter 7.

The hematology application revealed a problem that was not anticipated before. It turned out to be impossible to define forms that consisted of a combination of previously entered clinical information and new data elements. This problem was caused by the decision to mimic paper-based forms, leading to the assumption that information was only viewed and entered as a set of data elements combined in these forms. This was implemented in the software by defining forms as sets of data elements with two modes: display and data entry (i.e. the values of all data elements were either displayed or could be entered). Although it was theoretically possible to compose different forms that share data elements, it was not implemented. Two issues surfaced here: a mix of display modes and the management of updated values. Without going into too much detail, both issues are addressed here to demonstrate the underlying problems.
To allow mixed display modes, where some items are editable and some are not, the form definition needs to be expanded to allow a display mode setting. The presentation layer should be adjusted to honor the display mode setting. Retrieval of data for display might be subject to a time range. If items are too old for reliable display, the system should not display the items. This requires a change in the retrieval and display of the information, but also a change in the form definition and the presentation layer if the time range is selectable.
More importantly is the issue of updated value management. If an item is displayed in a form, a reference to the exact version of that item should be stored for future retrieval to avoid integrity problems when the item is updated in the mean time. The system should also notify the user that a more recent version of that particular item is available.
These issues can be partly resolved by a more generic presentation layer as described in chapter 8.

The clinical trial application showed that although it was easy to add a function to the form that performs calculations based on the values of the data elements, it would require a developer with considerable programming experience to correctly write the additional code. The calculation of the inclusion/exclusion criteria for this application was quite complex, but even more simple calculations such as a BMI would still need a programmer to develop the code.
A part of this complexity could be solved by including calculation algorithms in the archetype definition. These algorithms would be defined in an expression language that can be easily converted to executable code, yet can be understood and used by clinical users to define the algorithm. Part of the expression language is the definition of references to the actual values of the relevant parts of the archetype or even parts of other archetypes (e.g. the weight and height values are part of different archetypes than the BMI calculation). Another challenge is the handling of missing or outdated values.
Efforts in that direction are now undertaken by the *open*EHR Foundation [9].

Functional requirements were derived from literature and "expectations". Hence, not all required functionalities were foreseen. Currently, standards exist (or are in development) that provide lists of functional and EHR requirements. Such standards would now be a good starting point for the selection of functionality in a new general PropeRWeb implementation. [24, 25]

7.6 TOOLS

As stated before, the development speed of the applications would increase with the availability of appropriate tools. At the start of the PropeRWeb development, an early version of an archetype editor was available from the *open*EHR project. However, as the underlying reference model was subject to change and the output of the archetype editor produced Eiffel classes, the software was not useful for the PropeRWeb application. A similar editor would be necessary, that would provide XML documents.
An editor that could create the CForms definition files would also speed up the development. A combination of the two editors could provide a link between the archetype and the forms and their presentation layer counterparts. This issue is discussed in more detail in chapter 8.

7.7 DISCUSSION

Open source projects such as OpenEMed and Cocoon are usually started to solve the immediate needs of the initial developers. Both PIDS and COAS were not full implementations of the OMG specifications, but only the parts that were useful

for the project team. When an open source community grows, additional needs of new team members add more functionality to the code thus leading to a full implementation of the specifications. In this respect, the OpenEMed and Cocoon communities are quite different. Not only the size, but also the origin is different. The OpenEMed community has a few hundred members, while the Cocoon community has a few thousand members. The OpenEMed project implements standards, while the Cocoon project was started to solve a problem, complying with standards when relevant. The larger the community, the quicker a project grows in functionality.

The major advantage of open source is the access to the source code, which allows verification and extension of the implementation, as occurred with the PatientRelation trait. In contrast, a closed source model led to the failure to implement a connection to the various systems used by the CVA trial users because of the defensive position of the vendors.

The flexibility of a desktop application with regard to data entry cannot be met by a simple web application. By adding JavaScript functions to a page, a step towards the flexibility of a desktop application was made, but it was not until AJAX [26], a framework to optimize performance and increase user friendliness of web applications, was created in 2005 that the gap between desktop and web applications began to close. With the current AJAX based frameworks a new type of web applications, Rich Internet Applications (RIAs) [27] are now emerging and trying to combine the advantages of both desktop and web applications.

8 Conclusions

The key findings and challenges will be summarized here.

8.1 FINDINGS

8.1.1 Architecture

- The componentized architecture provided a stable foundation for a generic EHR system.
- PIDS was implemented well in the OpenEMed component and worked well in the PropeRWeb system. It demonstrated the necessity of a demographic service.

- COAS extended with the OpenEMed create functionality served its purpose in PropeRWeb, but might be replaced by other structures. It demonstrated the necessity for a service for storage and retrieval of clinical data.
- Archetypes are better suited for concept structuring than COAS ObsData structures.
- Separation of domain model and presentation layer revealed the need for a generic GUI and additional specifications to define presentation information in a generic way.
- The evaluation of PropeRWeb has demonstrated that a successful implementation of a generic, domain independent, EHR system is achievable.
- Archetype reusability was demonstrated by the reuse both within and across applications.
- An expression language should be developed to define calculations that are part of archetype definitions.
- Security should be integrated early in the design.
- Security that focuses on hiding unauthorized functionality has an impact on the GUI and therefore provides a challenge for a generic GUI.
- Web applications have several advantages such as platform independence and ubiquitous availability. Recent trends such as RIAs eliminate many of the disadvantages.

8.1.2 Implementation/Usability

- Version 2 of PropeRWeb was successful in solving the usability issues and the part of the performance issue.
- Version 2 also contributed in the separation of concerns, which in turn contributed to the ease of reconfiguration.
- Evaluation of the trial of version 1 showed that the users were in general satisfied with the system, despite the technical problems.
- The use of open source is advantageous not only because of the low cost, but because of the insight in the code and therefore the opportunity to verify the implementation and the possibility to add additional code.
- Expanding the functionality of open source applications requires extensive insight in the existing code and hence often has a steep learning curve.

8.2 CHALLENGES

Table 4 summarizes the key challenges, their impact and the action taken to minimize the impact.

Multiple changes were happening in several areas of the project: there were technical problems, changes in the care process, changes of key persons, introduction

TABLE 4 Key challenges

Finding/Challenge	Impact on the evaluation	Action taken
The JSP version (version 1) showed some negative performance and usability issues.	This contributed to the fact that the users reverted to the use of email as a means of communication.	Version 2, based on Apache Cocoon, was developed.
Connections to discipline dependent systems were not available.	This contributed to the fact that the users were reluctant to invest much effort in the system.	A print function was developed to substitute the relevant paper forms.
The GPRS connection was not as reliable as anticipated.	This contributed to the fact that the users reverted to email as a means of communication.	The users were informed on how to access the system from their own computers with a broadband Internet connection. This solved most performance problems.
The partnership with the clinical CVA project resulted in unanticipated dependencies.	The change of key persons resulted in less cooperation to provide information to the system.	A care coordinator agreed to enter the discharge information in the system.
(idem)	The change of the triage of the patients resulted in fewer patients for inclusion in the system.	The trial period was extended to allow inclusion of more patients.
External organizational changes were forced onto the users, such as the introduction of yet another registration system.	This doubled the required effort of the care coordinator.	The care coordinator was reimbursed for the extra hours.

of standards and guidelines by third parties, all leading to a failure to meet the original goal.

One avoidable problem was the omission to bind all parties involved in a formal way. The initial persons were all enthusiastic, but over time, most of them left the project due to other engagements. Their successors had different views and didn't feel obliged to participate in the same way.

Also, a more agile development process would have increased user involvement. To be really effective, the technologies to be used should be well known as to be able to meet the requirement of short development and feedback cycles. In a research project, such short cycles may be difficult to achieve.

There were technical problems, but more important for success or failure are the organizational issues. Without support and active participation, the chances of success are slim.

Nonetheless, the objective to design a generic EHR system was met. As demonstrated, the system could be configured for various domains without great effort.

This success can also be contributed to the use of archetypes, which were easily configurable and reusable. Their evolution over time has solved various issues that were encountered during implementation such as the terminology section in the archetype definition and the templates.

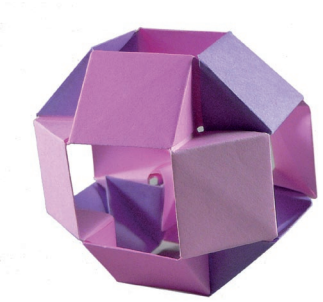
9 References

1 Werkgroep Thuiszorgteam. Het Transmuraal Zorgmodel CVA, Het Thuiszorgteam: Gestructureerde revalidatie in de Thuissituatie. Maastricht; 2000.
2 Tange H, Van Der Linden H, Sas P, Beusmans G, Talmon J, Van Oosterhout E, et al. Towards a PropeR combination of patient records and protocols. Int J Med Inform. 2003 Jul;70(2-3):141-8.
3 Commissie CVA-Revalidatie. Revalidatie na een beroerte: Dutch Heart Foundation; 2001.
4 OpenEMed. [cited 2009/01/21]; Available from: http://www.openemed.org/
5 Person Identification Service, version 1.1. [cited 2009/01/21]; Available from: http://www.omg.org/technology/documents/formal/person_identification_service.htm
6 Clinical Observations Access Service, version 1.0. [cited 2009/01/21]; Available from: http://www.omg.org/technology/documents/formal/clinical_observation_access_service.htm
7 HSQLDB. [cited 2009/01/21]; Available from: http://hsqldb.org/
8 Beale T. Archetypes and the EHR. Stud Health Technol Inform. 2003;96:238-44.
9 Beale T, Heard S. Archetype Definition Language, ADL 1.4. 2007 [cited 2009/01/21]; Available from: http://www.openehr.org/svn/specification/TRUNK/publishing/architecture/am/adl.pdf
10 Apache Cocoon. [cited 2009/01/21]; Available from: http://cocoon.apache.org/
11 Cocoon Forms, Introduction. [cited 2009/01/21]; Available from: http://cocoon.apache.org/2.1/userdocs/basics/index.html
12 Cocoon Template. [cited 2009/01/21]; Available from: http://cocoon.apache.org/2.2/blocks/template/1.0/1215_1_1.html
13 Corba Security Service. 2003 [cited 2009/01/21]; Available from: http://www.omg.org/technology/documents/formal/omg_security.htm#Security_Service
14 SOAP Version 1.2 Part 1: Messaging Framework (Second Edition). [cited 2009/01/21]; Available from: http://www.w3.org/TR/soap12-part1/
15 Web Services Activity. [cited 2009/01/21]; Available from: http://www.w3.org/2002/ws/Activity
16 openEHR Template. [cited 2009/01/21]; Available from: http://www.openehr.org/131-OE.html
17 Document Object Model (DOM). [cited 2009/01/21]; Available from: http://www.w3.org/DOM/
18 About SAX. [cited 2009/01/21]; Available from: http://www.saxproject.org/
19 Separation of concerns. [cited 2009/01/21]; Available from: http://en.wikipedia.org/wiki/Separation_of_concerns
20 van der Linden H, Diepen S, Boers G, Tange H, Talmon J. Towards a Generic Connection of EHR and DSS. Stud Health Technol Inform. 2005;116:211-6.
21 Ocean Informatics. [cited 2009/01/21]; Available from: http://www.oceaninformatics.com/
22 openEHR. [cited 2009/01/21]; Available from: http://www.openehr.org/home.html
23 Template Designer. [cited 2009/01/21]; Available from: http://www.oceaninformatics.com/Solutions/ocean-products/Clinical-Modelling/ocean-template-designer.html
24 ANSI/HL7 EHR System Functional Model, Release 1. 2007 [cited; Available from: http://webstore.ansi.org/FreeStandards.aspx?sku=ANSI/HL7%20EHR,%20R1-2007&sdo=HL7&deptid=3104&filename=ANSI+HL7+EHR,+R1-2007.zip&name=HL7%20EHR%20System%20Functional%20Model,%20Release%201
25 ANSI. ISO/TS 18308 Health Informatics - Requirements for an Electronic Health Record Architecture: ISO; 2003.
26 Garrett JJ. Ajax: A New Approach to Web Applications. 2005 [cited 2009/01/21]; Available from:

http://www.adaptivepath.com/ideas/essays/archives/000385.php
27 Rich Internet application. 2009 [cited 2009/01/21]; Available from: http://en.wikipedia.org/wiki/Rich_Internet_application
28 I18N Transformer. [cited 2009/01/21]; Available from: http://cocoon.apache.org/2.1/userdocs/i18nTransformer.html

Chapter 6

Trends in EHR System Architectures



Trends in EHR System Architectures

In this chapter, the validity of the PropeRWeb architecture is studied. The choices for this architecture as presented in chapters 2 and 3 are compared with the literature of 2004-2007; this period starting at the time when the PropeRWeb architecture was piloted in the CVA domain. The study addresses four research questions, which are related to the four areas of the literature study of chapter 2:

- Which relevant standards are currently used?
- Which EHR system architectures are described and how do they match the PropeRWeb architecture?
- Which requirements are added to the list we found in chapter 2?
- Which developments have taken place with regard to user interfaces?

The literature review for this study was conducted with a methodology similar to that reported in chapter 2. It resulted in 49 relevant articles. Table 5 shows a breakdown of the articles over the four categories.

1 Results

A general trend apparent from the papers is the increased focus on interoperability. The penetration of the Internet and the World Wide Web into everyday life provides the basic infrastructure for connecting systems. This is reflected in the papers, which have a more prominent focus on the issues involved in information exchange between systems.

TABLE 5 Division of articles in categories

Category	Initial screening	After consensus	Articles with relevant information
Standards	15	7	4
Requirements	37	16	7
Architecture	111	51	29
User Interface	27	16	9
Total	190	90	49

1.1 STANDARDS

The category Standards has four articles [1-4]. In the previous literature review, one prestandard on EHR system architecture was found, the Healthcare Information Systems Architecture (HISA) registered as CEN ENV 12967 [5]. Also, several standards for data representation or EHR data model architecture were found. The most relevant are the Electronic Health Record Communication (EHRCom, CEN-EN 13606) [6] and the HL7 Clinical Document Architecture (CDA). [7].

1.1.1 Architectural standards

The HISA prestandard became a full standard in 2007 [8]. The Distributed Healthcare Environment (DHE), a commercial HISA implementation, is in use in larger projects, such as a hospital information system in Denmark [9].

In 2007, the HL7 EHR System (EHR-S) Functional Model, Release 1 specifications were ANSI approved [10]. These specifications provide requirements specified as a standardized list of functions that may be present in an EHR system. "The function list is described from a user perspective with the intent to enable consistent expression of system functionality." [10] The model is divided in three sections: Direct Care, Supportive and Information Infrastructure. Examples of each section are respectively: Care Management, Administration & Financial and Security. A revision is underway and is expected to be open for comments and voting in 2009 in both HL7 and ISO.

Another standard on requirements, the ISO 18308 standard (Requirements for an Electronic Health Record Reference Architecture), is currently also under revision [11].

HISA is of less relevance for the PropeRWeb architecture. It provides a framework for integration of various IT systems in health care organizations of which the EHR system may be one. Of more importance are the HL7 EHR system functional model and the ISO 18308 standard. The HL7 standard specifies functional system requirements that could be implemented in the PropeRWeb system. These functions could be added to PropeRWeb in an incremental way, dependent on the user needs. The requirements of 18308 are important for the architecture of the EHR record. These requirements should in principle be reflected in the EHR record standards like the ISO 13606 and the HL7v3 RIM and CDA standard. The consequences for the PropeRWeb architecture are described in the next subsection.

In reflection, both the HL7 EHR system functional model release 1 and the ISO 18308 standard would have been of value had they been available at the start of our developments. They could have been used to specify a more solid EHR archi-

tecture as well as would allow the users of PropeRWeb to prioritize the functional system requirements to be available in the CVA system.

1.1.2 Clinical data representation and messaging

As said in chapter 2 there are three important standards on EHR data architecture: CEN EN13606 [12], HL7 CDA [1] and HL7 v3 Messaging Framework [13].
These standards, in draft then, have now been accepted (HL7 CDA R2 in 2005, CEN EN 13606 in 2006 and ISO 13606 in 2008) and several projects show implementations. *open*EHR archetypes [14] are also gaining momentum, not only in the development of a larger repository of available archetypes, but also by the number of projects that implement archetypes in an EHR system. These projects will be discussed in § 1.3.5.
Messages are used, not only to exchange information between systems, but also to exchange information between layers, such as between the application layer and the database layer [15]. In PropeRWeb, HL7v2 encodings are used in the communication with the PIDS and COAS server. The data are represented as traits (PIDS) and ObsData structures (COAS). These are rather limited representations of the demographic and clinical data. ISO/CEN 13606 and HL7v3 RIM allow for richer data structures. Currently, work is underway mainly in HL7 to define detailed clinical models, an approach to model clinical concepts in a Reference Model agnostic way.
Chapter 5 demonstrates that the approach in PropeRWeb for the representation of clinical concepts is usable. However, when it comes to data exchange with various other systems richer models are needed that include more contextual information. A redesign based on 13606 archetypes or on detailed clinical models is necessary to make PropeRWeb an application that can be used on a routine basis in a larger health care environment. It will enhance reuse of work done in the domain of clinical concept modeling and will also enhance the exchange of clinical data between various organizations.

1.1.3 Healthcare Domain Taskforce specifications

The CorbaMed or Healthcare Domain Taskforce (HDTF) specifications that were developed around 1991 have not seen major updates since. The website now states that the HDTF and HL7 have joined forces in the Healthcare Services Specification Project (HSSP) [16].
This project is intended to "identify and document service specifications, functionality, and conformance supportive and relevant to healthcare IT stakeholders and resulting in real-world implementations. In addition, several other groups have joined the HSSP effort." [17]. The HSSP project focuses on more technical specifications whereas the HL7 EHR-S specifications focus on a higher, more abstract level.

The Person Identification Service (PIDS) [18] is in itself not used very often, although it has been implemented in at least one commercial product [19]. However, several projects have incorporated the concept by designing different databases for demographic and clinical data [20-22] or by implementing a functionality of mapping of Patient IDs in a broker (e.g. [23]), which resembles the Correlation Manager of the PIDS specification.
The Clinical Observation Access Service (COAS) [24] has not been implemented in the projects studied in this review. However, most have adopted some form of clinical repository. The strength of COAS was the flexibility of its data structure that allows storage of virtually any data. This concept is incorporated in the standards mentioned above, i.e. CEN EN 13606, HL7 CDA, HL7 v3 Messages and archetypes.
Implementation of the Resource Access Decision Framework (RAD) component [25] is sparsely reported in the literature, but the intention, a separate service for access control, is a prominent part of every architecture of larger systems.
The Lexicon or Terminology Query Service (LQS/TQS) [26] was designed to provide a generic access to underlying Terminology Services. Literature shows that terminology and ontologies are becoming more important [22], but remain at the level of proprietary implementations. The policy change to make SNOMED-CT widely available at a low cost may change this picture.

In conclusion, it can be said that the HDTF specifications have failed to survive as components, but have provided concepts for the implementation of distinct services and have moved the idea of a componentized architecture forward.
In general, the standards emerging in the time frame discussed in chapter 2 have evolved and matured without major changes that would invalidate the concepts of the PropeRWeb framework.

1.2 REQUIREMENTS

The Requirements category consists of seven articles [27-33]. The list of requirements found in the previous literature review in chapter 2 remains valid to date. The experience gained in recent research has added requirements to ensure connections between EHR systems. Explicitly by focusing on semantic interoperability [2, 31] and requirements for a virtual, cross-organizational system [33]. Implicitly by focusing on national patient IDs [29].
These additions fit the general trend towards information integration for the purpose of a virtual patient record. These requirements have not been implemented in the PropeRWeb framework. The underlying security issues that occur in cross-organizational systems are the subject of study as described in the next chapter 7.

1.3 ARCHITECTURE

The category Architecture has 29 articles, 27 architectural descriptions and 2 review articles on architectural approaches [9, 15, 20-23, 34-56].

The architectural descriptions show several commonalities:

- Layers
- Scope
- Web oriented
- Open source
- Two-model approach

The articles show a trend not only towards connection with other systems, but also to usage of the available functionalities of the Internet. Another significant trend is the use of open source software.

1.3.1 Layers

The software is divided in layers. The most common division is database, application and presentation layer as shown in [9, 41, 45, 47, 49, 56] amongst others. Only a few papers give a detailed description of the content of the various layers. These papers present components that provide dedicated services such as patient identification and care-oriented services. Such a layered approach is also found in PropeRWeb. This proves to be an approach that is shared by others.

1.3.2 Scope or inter-organizational connections

The literature shows a trend towards connections across organizational borders [21, 22, 34, 38, 40, 50, 51, 55]. The approach taken is either a federated approach [20, 38, 39, 51-53] or a broker approach [15, 22, 23, 35, 40, 52, 54]. In a federated approach, information from distributed systems is centrally stored, whereas a broker has a minimal set of information to allow retrieval from the distributed systems itself. Well-known analogies are the proxy server and the web search engine: a proxy server caches frequently used web pages, while a web search engine contains a reference to a web page as well as keywords related to the content of the web page, but not the page itself.

Two alternatives are presented:

- The Indivo project (formerly known as PING) focused on a patient controlled EHR. Patients not only control who has access to the embedded information, but can also specifically request healthcare professionals to put information into the system. [41] They implemented this concept in a working system.
- The Swiss project described by Geissbuhler et al [43] that focuses on a regional

infrastructure in which the connected systems are equals, i.e. a peer-to-peer network. Although there is no broker in the usual sense, there is a lightweight broker that contains a central repository of listed healthcare professionals and their access rights and a list of connected mediators.

- Also in other countries developments take place that promote the exchange of information. In the USA, several Health Information Exchanges are being developed using a variety of approaches. In the Netherlands a national infrastructure is being developed largely following the broker approach.
- Although not studied in our project, inter-organizational connections are in principle possible through the COAS server approach of PropeRWeb. When clinical data in various systems are accessible by COAS services, the PropeRWeb architecture allows access to such resources.

1.3.3 Web oriented architecture

In this literature review, there are very few references to CORBA. CORBA is generally considered complex [57] and slow. Di Giacomo et al. [21] have demonstrated that CORBA is too slow for quick data retrieval.

The expanding scope towards regional systems makes web services more practical to use than CORBA, because the aspect of "loose coupling" is easier to implement.

At the time the development of PropeRWeb started, web services hardly existed. Security was poorly handled, so CORBA seemed the way to go at that time. In addition, the reference implementations of PIDS and COAS were CORBA based although the HDTF specifications only specify the services, not the way they are addressed.

A change towards web services mainly affects the lower levels of the architecture. Only the interface in the middleware that connects the business logic with the databases needs to be redesigned. The basic principles of PropeRWeb are not affected by this evolution of web based intersystem information exchange.

1.3.4 Open source software

Many papers describe the use of open source software as the basis of their project. In some cases, they only use open source components but keep their own contributions closed source [50]. In other cases, the entire project is open source.

The benefit of using open source is not only the reduced costs, but also the quality of the software. A frequently used set of open source projects (Linux, the Apache web server, PHP and MySQL, also referred to as LAMP) provides an industry-strength environment for the development of web-based projects. Unfortunately, the projects usually do not explain their choice for open source.

The use of Java, although strictly speaking not open source, is another trend. Java

provides platform independency and web based functionality. The papers in this category showed at least 10 projects in which Java was used [35, 38, 39, 41, 44, 48-50, 55, 56]
The use of XML is widespread, ranging from configuration to XSL transformation of data structures for integration.

Two examples of open source projects are Indivo [41], mentioned earlier, and OpenMRS [48].
OpenMRS is an open source project that resembles the PropeRWeb system in its purpose of a patient-oriented EHR system and its ease of reconfiguration for different domains. It differs from PropeRWeb in the definition of the concepts. OpenMRS uses a concept dictionary whereas PropeRWeb is based on archetypes. OpenMRS is successfully implemented in several developing countries.
Despite the steep learning curves for the CORBA based HDTF services and the Cocoon framework, the use of open source components has helped in the development of PropeRWeb. The creation of the PIDS and COAS services surely would have required more time.
The development of open source implementations of standards for EHR functionalities or services will help the dissemination and adoption of the standards. It furthermore provides a reference implementation that demonstrates how the standard should (or could) be read.

1.3.5 Dual model approach

The dual model or two-model approach, as explained in § 3.1.4 of chapter 3, to an EHR system is said to provide a strong future-proof system. This is one of the reasons for implementing this approach in the PropeRWeb framework by using archetypes.
Muñoz et al [35] describe a proof-of-concept system based on EN 13606 and archetypes. They concluded that the system was useful in real-time settings (after solving an unrelated issue). The EN13606 provided reliable support for the exchange of information and the system was used in different scenarios (i.e. domains) without modifications other than a change in the archetypes.
The Julius system, by Chen et al. [56], is similar to PropeRWeb in that it follows a dual model approach to create a generic system that can integrate information from various EHR systems. This system uses archetype-like constructs, called templates. The approach taken in PropeRWeb to separate the clinical model from the informatics model has proven to be a valid one. Not only was it demonstrated in our evaluation that PropeRWeb could easily be tuned to the needs of other domains (see chapter 5), the current literature supports the two-model approach. Also the

work in ISO on detailed clinical models demonstrates the need for the separation of the data base (or reference information) model from the clinical concepts.

1.3.6 IHE

Although there is only one explicit reference in the literature studied [33], the IHE initiative is important to the topic of this chapter.

Integrating the Healthcare Enterprise (IHE) is an initiative to improve information sharing between computer systems in health care. IHE is formed by a consortium of radiologists and information technology experts. They try to close the gap between standards and implementation. [58] This is achieved by the definition of IHE Profiles. These Profiles provide a solution to a specific, usually domain-specific, problem. Implementation of these Profiles is described in Technical Frameworks. Since the initiative focuses on vendors and users of larger health care systems, the solution results by definition in a componentized architecture.

1.4 USER INTERFACE

The category User interface has nine articles [59-67]. The articles focus on the best user interface to optimally support the user's workflow. There is a trend to web-based GUIs, which is in line with the trends seen in the other categories.

A few articles focus on the design of a GUI or the redesign of a failing UI. [62, 65]. Their main conclusion is that good user interfaces contribute to improve the quality of care and reduce errors. To achieve good user interfaces, they should be based on user-centered design principles.

A GUI needs to be flexible, easily adaptable by either the user or the developer. This can be achieved by generating the GUI, rather than predefining the GUI at build time. [60, 61, 64]

Marrying these two conclusions can be a challenge, which is addressed in chapter 8.

The trend of web-based GUIs supports the decision of the web-based implementation of the PropeRWeb system.

2 General discussion

The health informatics field is still moving and shows a movement towards services and components. Although the HL7 EHR-S Functional Model is not intended to be an implementation specification, its division into services supports a componentized architecture.

The widening scope shows a trend towards regionally connected systems, which, due to their distributed nature, already follow a componentized architecture. In such a regional system, the underlying systems can be considered as components or data sources. This also requires standards to ensure semantic interoperability between the various connected systems.
The large number of web-based projects shows a trend towards ubiquitous computing. It doesn't matter which operating system the user uses, or which browser. The Internet provides a generic way to access the information independent of the time and of the location of the user.
Smaller projects, which are implementing more patient-oriented systems, such as Indivo and OpenMRS show a trend to the use of open source software. Not only by reusing the results of available open source projects, but also by making their own source code available. This allows a more efficient use of software and developer resources than the common closed source projects which tend to reinvent the wheel.

From this literature review it can concluded that the componentized PropeRWeb architecture based on a two-model approach has not become common practice in EHR systems implemented within an organization. However, projects that work on systems on a regional or national level do follow the same componentized approach. Several projects demonstrate the validity and strength of the two-model approach. As with PropeRWeb, they provide a certain amount of domain independence, i.e. it is possible to configure the system for a different domain, without any changes or with just small changes to the system itself.

It can be concluded that the design principles of PropeRWeb are still valid. At the implementation level, other choices can be made, most notably with respect to the use of web services rather than CORBA. Secondly, the archetypes approach should be made compliant with the specifications in the ISO 13606 standard and the *open*EHR approach.

3 References

1 Dolin RH, Alschuler L, Boyer SL, Beebe C, Behlen FM, Biron PV, et al. HL7 Clinical Document Architecture, Release 2. J Am Med Inform Assoc. 2006 Jan 1;13(1):30-9.
2 Kalra D. Electronic health record standards. Yearbook of medical informatics. 2006 Jan 1:136-44.
3 Jian WS, Hsu CY, Hao TH, Wen HC, Hsu MH, Lee YL, et al. Building a portable data and information interoperability infrastructure-framework for a standard Taiwan Electronic Medical Record Template. Comput Methods Programs Biomed. 2007 Nov 1;88(2):102-11.

4 Ferranti JM, Musser RC, Kawamoto K, Hammond WE. The clinical document architecture and the continuity of care record: a critical analysis. J Am Med Inform Assoc. 2006 May-Jun;13(3):245-52.
5 Ferrara FM. The standard 'Healthcare Information Systems Architecture' and the DHE middleware. Int J Med Inform. 1998 Oct 1;52(1-3):39-51.
6 Markwell D, Fogarty L, Hinchley A. Validation of a European message standard for electronic health records. Stud Health Technol Inform. 1999;68:818-23.
7 Dolin RH, Alschuler L, Beebe C, Biron PV, Boyer SL, Essin DJ, et al. The HL7 Clinical Document Architecture. J Am Med Inform Assoc. 2001 Nov 1;8(6):552-69.
8 Medische informatica - Services architectuur - Deel 1: Ondernemingsstandpunt. 2007 [cited 2009/01/26]; Available from: http://www2.nen.nl/nen/servlet/dispatcher.Dispatcher?id=BIBLIOGRAFISCHEGEGEVENS&contentID=248200
9 Bernstein K, Bruun-Rasmussen M, Vingtoft S, Andersen SK, Nohr C. Modelling and implementing electronic health records in Denmark. Int J Med Inform. 2005 Mar 1;74(2-4):213-20.
10 ANSI/HL7 EHR System Functional Model, Release 1. 2007 [cited; Available from: http://webstore.ansi.org/FreeStandards.aspx?sku=ANSI/HL7%20EHR,%20R1-2007&sdo=HL7&deptid=3104&filename=ANSI+HL7+EHR,+R1-2007.zip&name=HL7%20EHR%20System%20Functional%20Model,%20Release%201
11 ANSI. ISO/TS 18308 Health Informatics - Requirements for an Electronic Health Record Architecture: ISO; 2003.
12 EN13606 - a Standard for EHR System Communication. [cited 2009/01/26]; Available from: http://www.openehr.org/standards/cen.html
13 Health Level 7 Version 3.0. [cited 2009/01/26]; Available from: http://www.hl7.org/Library/standards_non1.htm#HL7%20Version%203
14 openEHR. [cited 2009/01/21]; Available from: http://www.openehr.org/home.html
15 Cruz-Correia RJ, Vieira-Marques PM, Ferreira AM, Almeida FC, Wyatt JC, Costa-Pereira AM. Reviewing the integration of patient data: how systems are evolving in practice to meet patient needs. BMC Med Inform Decis Mak. 2007;7:14.
16 Healthcare DTF Home Page. [cited 2009/01/26]; Available from: http://healthcare.omg.org
17 Healthcare Services Specification Project (HSSP) Overview. [cited 2009/01/26]; Available from: http://hssp.wikispaces.com/
18 Person Identification Service, version 1.1. [cited 2009/01/21]; Available from: http://www.omg.org/technology/documents/formal/person_identification_service.htm
19 McKessonHBOC's HorizonWP Passport Wins Industry Award for Innovation. 2001 [cited 2009/02/03]; Available from: http://www.mckesson.com/en_us/McKesson.com/About+Us/Newsroom/Press+Releases+Archives/2001/McKessonHBOC%2527s+HorizonWP+Passport+Wins++Industry+Award+for+Innovation.html
20 Roman I, Roa LM, Reina-Tosina J, Madinabeitia G. Demographic management in a federated healthcare environment. Int J Med Inform. 2006 Sep 1;75(9):671-82.
21 Di Giacomo P, Ricci FL, Bocchi L. Integrated electronic health records management system. Stud Health Technol Inform. 2006;121:228-41.
22 Boniface M, Watkins ER, Saleh A, Dogac A, Eichelberg M. A secure semantic interoperability infrastructure for inter-enterprise sharing of electronic healthcare records. Stud Health Technol Inform. 2006;120:225-35.
23 Boyd AD, Hosner C, Hunscher DA, Athey BD, Clauw DJ, Green LA. An 'Honest Broker' mechanism to maintain privacy for patient care and academic medical research. Int J Med Inform. 2007 May 1;76(5-6):407-11.
24 Clinical Observations Access Service, version 1.0. [cited 2009/01/21]; Available from: http://www.omg.org/technology/documents/formal/clinical_observation_access_service.htm
25 Resource Access Decision, version 1.0. [cited 2009/01/26]; Available from: http://www.omg.org/technology/documents/formal/resource_access_decision.htm
26 Lexicon Query Service, version 1.0. [cited 2009/01/26]; Available from: http://www.omg.org/technology/documents/formal/lexicon_query_service.htm
27 Oie GR, Andresen H, Tondel IA. Handling consent to patient data access in a hospital setting. Medinfo. 2007;12(Pt 1):242-6.

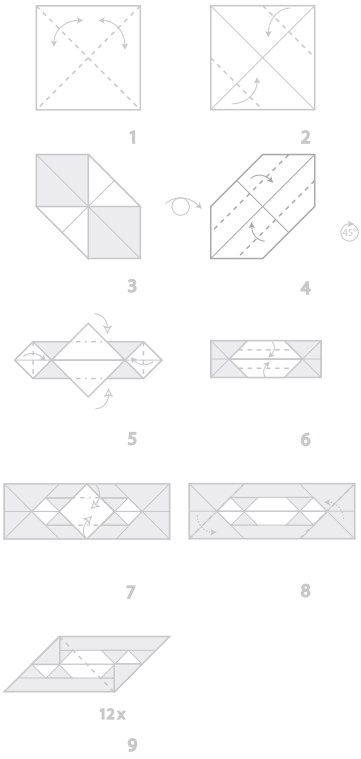
28 Spooner SA. Special requirements of electronic health record systems in pediatrics. Pediatrics. 2007 Mar 1;119(3):631-7.
29 Quantin C, Allaert FA, Avillach P, Riandey B, Fieschi M, Fassa M, et al. Proposal of a French health identification number interoperable at the European level. Medinfo. 2007;12(Pt 1):503-7.
30 Ruotsalainen P. Security requirements in EHR systems and archives. Stud Health Technol Inform. 2004;103:453-8.
31 Kalra D, Blobel BG. Semantic interoperability of EHR systems. Stud Health Technol Inform. 2007 Jan 1;127:231-45.
32 Hardiker NR, Bakken S. Requirements of tools and techniques to support the entry of structured nursing data. Medinfo. 2004;11(Pt 1):621-5.
33 Schabetsberger T, Ammenwerth E, Breu R, Hoerbst A, Goebel G, Penz R, et al. E-health approach to link-up actors in the health care system of Austria. Stud Health Technol Inform. 2006;124:415-20.
34 Andriuskevicius E, Dobravolskas E, Punys V, Sinciene V, Valentinavicius A. Architecture for national e-health infrastructure in Lithuania. Stud Health Technol Inform. 2006;124:421-6.
35 Munoz A, Somolinos R, Pascual M, Fragua JA, Gonzalez MA, Monteagudo JL, et al. Proof-of-concept design and development of an EN13606-based electronic health care record service. J Am Med Inform Assoc. 2007 Jan 1;14(1):118-29.
36 Zhang J, Sun J, Yang Y, Chen X, Meng L, Lian P. Web-based electronic patient records for collaborative medical applications. Comput Med Imaging Graph. 2005 Mar 1;29(2-3):115-24.
37 Antohi R, Ogescu C, Stefan L, Raureanu M, Onofriescu M, Toma M. Management of the electronic patient records in the web based platform for diagnosis and medical decision for optimization in healthcare-PROMED. Stud Health Technol Inform. 2007;127:98-107.
38 Bellika JG, Sue H, Bird L, Goodchild A, Hasvold T, Hartvigsen G. Properties of a federated epidemiology query system. Int J Med Inform. 2007 Sep 1;76(9):664-76.
39 Sucurovic S. Implementing security in a distributed web-based EHCR. Int J Med Inform. 2007 May 1;76(5-6):491-6.
40 Rigby MJ, Budgen D, Brereton OP, Bennett K, Layzell P, Keane J, et al. Proving the concept of a data broker as an emergent alternative to supra-enterprise EPR systems. Med Inform Internet Med. 2005 Jun 1;30(2):99-106.
41 Mandl KD, Simons WW, Crawford WC, Abbett JM. Indivo: a personally controlled health record for health information exchange and communication. BMC Med Inform Decis Mak. 2007;7:25.
42 Blobel BG. Advanced and secure architectural EHR approaches. Int J Med Inform. 2006 Mar 1;75(3-4):185-90.
43 Geissbuhler AJ, Spahni S, Assimacopoulos A, Raetzo MA, Gobet G. Design of a patient-centered, multi-institutional healthcare information network using peer-to-peer communication in a highly distributed architecture. Medinfo. 2004;11(Pt 2):1048-52.
44 Kardas G, Tunali ET. Design and implementation of a smart card based healthcare information system. Comput Methods Programs Biomed. 2006 Jan 1;81(1):66-78.
45 Spidlen J, Hanzlicek P, Riha A, Zvarova J. Flexible information storage in MUDR(II) EHR. Int J Med Inform. 2006 Mar 1;75(3-4):201-8.
46 Crespo Molina P, Angulo Fernandez C, Maldonado Segura JA, Moner Cano D, Robles Viejo M. Non-invasive light-weight integration engine for building EHR from autonomous distributed systems. Stud Health Technol Inform. 2006;124:173-8.
47 Simons WW, Mandl KD, Kohane IS. The PING personally controlled electronic medical record system: technical architecture. J Am Med Inform Assoc. 2005 Jan 1;12(1):47-54.
48 Allen C, Jazayeri D, Miranda J, Biondich PG, Mamlin BW, Wolfe BA, et al. Experience in implementing the OpenMRS medical record system to support HIV treatment in Rwanda. Medinfo. 2007;12(Pt 1):382-6.
49 Hanzlicek P, Spidlen J, Nagy M. Universal electronic health record MUDR. Stud Health Technol Inform. 2004;105:190-201.
50 Spitzer M, Brinkmann L, Ueckert FK. Clearinghouse: a teleradiology platform emphasizing security of data and communication. Medinfo. 2007;12(Pt 1):508-12.
51 McMurry AJ, Gilbert CA, Reis BY, Chueh HC, Kohane IS, Mandl KD. A self-scaling, distributed infor-

mation architecture for public health, research, and clinical care. J Am Med Inform Assoc. 2007 Jul 1;14(4):527-33.
52 Mykkanen J, Riekkinen A, Sormunen M, Karhunen H, Laitinen P. Designing web services in health information systems: from process to application level. Int J Med Inform. 2007 Feb 1;76(2-3):89-95.
53 Bergmann J, Bott OJ, Pretschner DP, Haux R. An e-consent-based shared EHR system architecture for integrated healthcare networks. Int J Med Inform. 2007 Feb 1;76(2-3):130-6.
54 Curtis AC, Gillon J, Malmrose DC. Integration of longitudinal electronic records in a large healthcare enterprise: the U.S. Veterans Health Administration experience. Medinfo. 2007;12(Pt 1):367-71.
55 Poulymenopoulou M, Vassilacopoulos G. An electronic patient record implementation using clinical document architecture. Stud Health Technol Inform. 2004 Jan 1;103:50-7.
56 Chen R, Enberg G, Klein GO. Julius–a template based supplementary electronic health record system. BMC Med Inform Decis Mak. 2007;7:10.
57 Common Object Request Broker Architecture. [cited 2009/01/26]; Available from: http://en.wikipedia.org/wiki/Common_Object_Request_Broker_Architecture
58 Integrating the Healthcare Enterprise. [cited 2009/01/26]; Available from: http://www.ihe.net/
59 Sundvall E, Nystrom M, Forss M, Chen R, Petersson H, Ahlfeldt H. Graphical overview and navigation of electronic health records in a prototyping environment using Google Earth and openEHR archetypes. Medinfo. 2007;12(Pt 2):1043-7.
60 Schuler T, Boeker M, Klar R, Muller ML. A generic, web-based clinical information system architecture using HL7 CDA: successful implementation in dermatological routine care. Medinfo. 2007;12(Pt 1):439-43.
61 Schuler T, Garde S, Heard S, Beale T. Towards automatically generating graphical user interfaces from openEHR archetypes. Stud Health Technol Inform. 2006;124:221-6.
62 Johnson CM, Johnson TR, Zhang N. A user-centered framework for redesigning health care interfaces. J Biomed Inform. 2005 Feb 1;38(1):75-87.
63 Hanzlicek P, Spidlen J, Heroutova H, Nagy M. User interface of MUDR electronic health record. Int J Med Inform. 2005 Mar 1;74(2-4):221-7.
64 Los RK, van Ginneken AM, van der Lei J. OpenSDE: a strategy for expressive and flexible structured data entry. Int J Med Inform. 2005 Jul 1;74(6):481-90.
65 Zheng K, Padman R, Johnson MP. User interface optimization for an electronic medical record system. Medinfo. 2007;12(Pt 2):1058-62.
66 Klimov D, Shahar Y. A framework for intelligent visualization of multiple time-oriented medical records. AMIA Annu Symp Proc. 2005:405-9.
67 Boye N, Eberholst F, Farlie R, Sorensen LB, Lyng KM. User driven, evidence based experimental design; a new method for interface design used to develop an interface for clinical overview of patient records. Stud Health Technol Inform. 2007;129(Pt 2):1053-7.

Chapter 7

Inter-organizational future-proof EHR systems

A review of the security and privacy related issues



Published as Helma van der Linden, Dipak Kalra, Jan Talmon, Inter-organizational future proof EHR systems, A review of the security and privacy related issues, International Journal for Medical Informatics, Vol 78, 2009, p 141-60. Epub 2008 Aug 28.

Inter-organizational future-proof EHR systems

1 Introduction

Most electronic medical record (EMR) systems that are currently in use are built and implemented with only local usage in mind. Communication between healthcare workers is translated to communication between systems and implemented as a one-to-one exchange of messages where the initiating party a priori knows the party to be queried (e.g. the physician entering a lab order in the EMR system which passes the order onto a known laboratory system and the lab returning the results to the EMR system). Such an exchange is comparable to a telephone call or its internet-equivalent: the email conversation. Tailoring the software to the specific exchange solves current interoperability issues such as differences in data structures and ambiguous interpretation due to implied metadata (e.g. absent measurement units or usage of terminology codes without specified coding schemes).
Problems arise when a new party requests access to the EMR information, the total number of users increases, the need for record sharing increases, and also when the clinical structure, organization and content need to evolve. These challenges require a generic interface that can comply with these changes, and mechanisms to find the location of the requested data.

Since the early years of this century the view has developed that high quality health care can be delivered only when all pertinent data on the health of a patient is available to the clinician. This changing point of view brings forth the notion of the (virtual) electronic health record (EHR) and requires ubiquitous communication between systems. Architectures have been designed to incorporate first generation interoperability standards, mainly in the 1990's, such as by the Synapses and Synex projects [1-3], but usually with the focus on a single organization or a single system available for multiple parties (e.g. a regional system). The HARP project ([4]) was the first to develop a more comprehensive componentized architecture that included a secure approach to clinical data sharing in a distributed environment. New-generation standards are better able to support elaborate communication between EMR systems. The most important are HL7 Version 3 [5] and CEN/TC215 13606 [6-10], which are now the subject of new implementation projects and technical evaluations.
Another project started by the *open*EHR Foundation is to develop an open, interoperable health-computing platform, of which a major component is clinically effective and interoperable EHR systems [11].

"The virtual EHR" is a term commonly used to denote the logical integration of distributed systems containing electronic medical or health records, irrespective of how this is achieved physically. Another commonly used term is "lifelong patient record", focusing on the availability of data and its integration over time. A lifelong virtual EHR (lvEHR) is currently seen as the best solution to meet the increasing demands for shared information described before in [12-14].

Consider the situation where an lvEHR integrates EMR systems from various healthcare organizations through the implementation of the new communication standards. Communication between these systems occurs based on information needs of the healthcare worker. The actual location of the information becomes transparent and of less importance.

Security and privacy related issues are more important in such an environment than in the current systems. For example, in the current situation access rights are defined locally, based on formal or less formal rules of the house. When dealing with access from outside the situation may ask for different requirements. In the current situation, outsiders can only have access to patient data through human intermediation, often the treating physician; she may act as a filter on what is communicated taking the patient's wishes into account. In a fully digital communication patient's wishes regarding disclosure of information have to be respected as well.

It is our objective to analyze these issues in the context of the lvEHR. We will also identify issues that require further study and regulations before safe and trustworthy lvEHR systems can become reality.

First we present the methods used, next we provide some definitions of concepts that are relevant in this article and the environment we assume in this article. Then we describe the results of our analysis, we discuss the results and present our conclusions.

2 Methods

Rather than approaching the problem of discovering and analyzing inter-organizational issues from a theoretical point of view, we have grounded the issues in a realistic clinical setting. We have used an imaginary, yet realistic scenario that focuses on shared care (see Appendix C). The scenario was divided into steps. Each step is an action that involves information exchange.

In several brainstorm sessions, HvdL and JT discussed each step from an implementation point of view (focusing on a global, rather than a more technical level). During these sessions questions were formulated that would reveal the issues

without giving a solution. Defining solutions for the issues raised in earlier questions could influence later questions, a situation we tried to avoid.
The questions were grouped together in themes and answers were discussed in more detail. The arguments were supported by references to relevant literature.
We verified that our question set was complete enough by mapping the themes to topics mentioned in various standards on security and privacy in EHRs.

3 Definitions

Although there is no consensus on the exact definition of the EHR, the ISO/TS 18308 [15] standard does give a definition of the primary purpose of the EHR:

> *The primary purpose of the EHR is to provide a documented record of care which supports present and future care by the same or other clinicians. This documentation provides a means of communication among clinicians contributing to the patient's care.*

A formal definition of the scope and purpose of the Integrated Care EHR has more recently been published as ISO TR 20514 [16].

> *"a repository of information regarding the health status of a subject of care in computer processable form, stored and transmitted securely, and accessible by multiple authorised users. It has a standardised or commonly agreed logical information model which is independent of EHR systems. Its primary purpose is the support of continuing, efficient and quality integrated health care and it contains information which is retrospective, concurrent and prospective."*

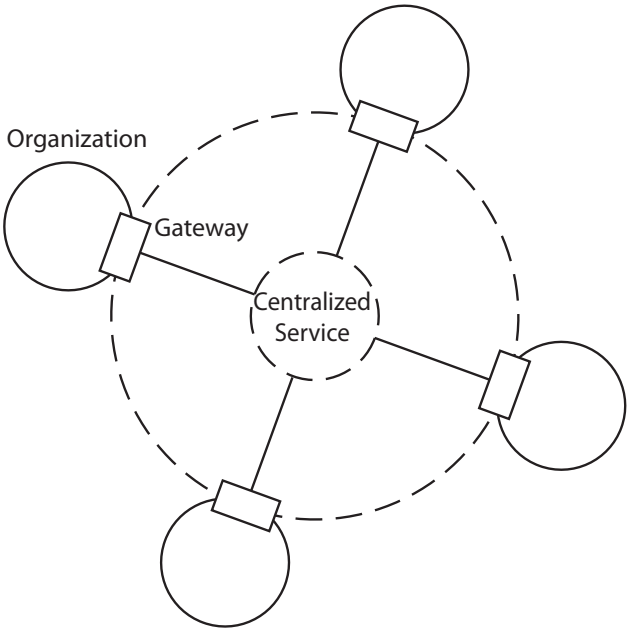
3.1 DEFINITION OF THE EHR INFORMATION

From the ISO definitions above we can infer that the total amount of health-related information about a patient that is stored in various systems, constitutes his (lifelong) (virtual) electronic health record. The healthcare worker who is involved in the care of a specific person will be referred to as the user in this article. This user needs access to the information in the EHR to provide adequate care to the patient.

This set of data is restricted by three partially overlapping divisions:

- *Origin of the information*: information is stored either in a local system or the information resides elsewhere (i.e. an external system). This is referred to as local and external information in this paper, respectively.

FIGURE 12 Cross-organizational environment.



- *Access*: several standards [9, 15] define access restrictions to protect the privacy of the patient. This is referred to as allowed and unallowed information.
- *Necessity or relevancy*: access to information is only allowed when it is relevant to the care process [17], again to preserve patient privacy. This is also known as the "need to know" principle.

From the viewpoint of the user, he is confronted with these three divisions: local information is readily available, while external information has to be retrieved from systems where he does not necessarily have direct access rights; he should be able to view information that is relevant to the current problem of the patient, but some of this information might be hidden due to access restrictions.
A true transparent lvEHR system would hide these divisions from the user. This allows the user to focus on his task while the system handles the burden of locating and retrieving information.

3.2 DEFINITION OF THE ENVIRONMENT

The environment described in the introduction resembles the configuration of Figure 12, in which multiple organizations are connected through a central service. Each organization has a gateway that hides the internal architecture from the outside world and vice versa.

The main purpose of the central service is to provide (a pointer to) the requested information to the (gateway of the) requesting EHR system(s).
This situation is comparable to an environment of distributed databases, with the difference that in a distributed database environment, the information is accessed on the database level (i.e. database schema and location are known beforehand), whereas in the environment described here the information is accessed on the application interface level (i.e. the application manages the actual information retrieval from the underlying database).

We have chosen this architecture as a basis for our discussion because it represents current best practice in a networked environment: a generic or standardized interface reduces the number of variations in implementations, while a central service reduces the number of possible connections to be made. Note that we have made no assumption about the primary location of the retrievable information, whether it remains in the respective systems or in the centralized service.

3.3 EHR REQUIREMENTS

Various authors have described generic requirements for EHR systems. We discuss only those related to security and privacy issues, with their definitions as stated in ISO/TS 18308 [15].

- *Security* There is consensus on the security requirements for EHR systems and for communication between systems [15]. We list the definitions of the most relevant security requirements:
  - *Authentication*: verifying the claimed identity of an entity (either a person or a system).
  - *Authorization*: granting rights, which include the granting of access based on access rights, either in a personal capacity or conferred on the basis of a role held by the user.
  - *Integrity (of data)*: preserving the accuracy and consistency of data regardless of changes made[9].
  - *Non-repudiation*: allowing any actor to obtain proof, which cannot be forged, that confirms the integrity and origin of a data item[10].
  - *Confidentiality*: the property of data that indicates the extent to which

9 Although this is a slightly paraphrased definition of the ISO standard, integrity should not be mixed with accuracy. While the latter deals with how well a data item represents reality, integrity deals with maintaining the proper representation of the recorded data item.

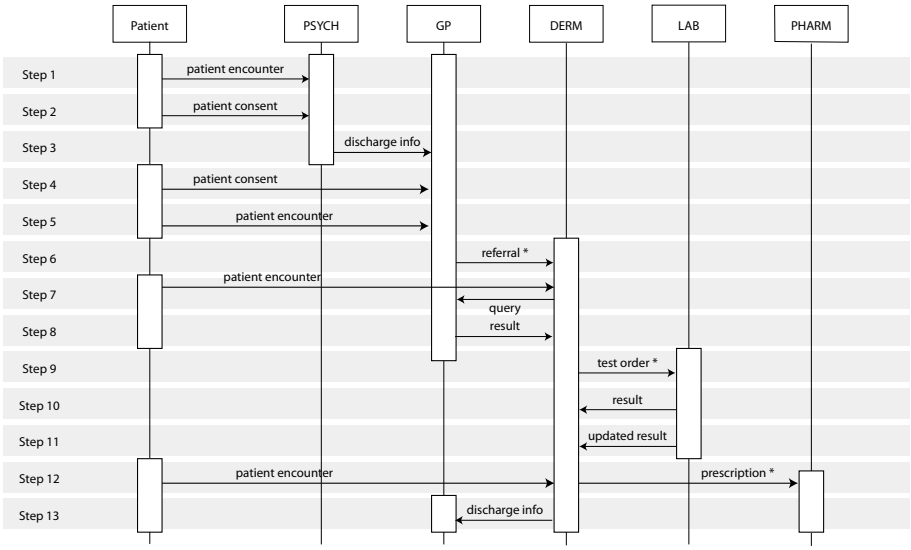
10 Although this requirement is often interpreted as being able to get proof of integrity and origin of the data that is communicated, full trustworthiness is only obtained when such proof can also be obtained on requests and receipt confirmations.

these data have not been made available or disclosed to unauthorized individuals, processes, or other entities.
    - *Consent*: obtaining, recording and tracking of informed consent of patients for the purpose of creating and allowing access to their health information for specified purposes during specified time frames[11].
- *Semantic interoperability*: enabling the ability to share data between systems that can be understood at the level of formally defined domain concepts to support automatic processing of data at the receiving system.
- *Author responsibility*: ensuring that each record contribution can be attributed to an identified author.
- *Audit trail or audit log*: recording activities of information system users in chronological order, which enables prior states of the information to be faithfully reconstructed. It should contain information about access to and modifications of data as well as the nature of each access and/or modification. It should also be able to support accountability for each interaction with the system by an actor (for example, logging each recorded step or task in the clinical or operational process).
- *Version management or version control*: supporting versioning at the granularity at which information is attested and supporting measures to discern modification or updating of the record.
- *Patient access*: allowing the patient access to all his EHR information subject to jurisdictional constraints.
- *Archiving & data retention*
    - *Data retention*: providing the functionality to store patient information for at least the duration specified in legal data retention policies. This also includes the capability of preventing deletion during the retention time and of insisting on deletion of information after that period [18].
    - *Archiving*: moving EHR information to off-line storage in a way that ensures the possibility of restoring them to on-line storage when needed without the loss of meaning.

These issues can often be managed quite simply when dealing with data residing in systems in a single organization, but for the IvEHR, these issues may have other dimensions and solutions may be less trivial. Perhaps the most challenging ones relate to achieving consensus on what, how and when different requirements should be implemented.

11 The ISO standard does not define the concept of consent. We inferred this definition from the descriptions of managing consent.

FIGURE 13 Sequence diagram of scenario. In this diagram the various columns represent the respective actors in the scenario (PSYCH = psychiatrist, GP = General Practitioner, DERM = Dermatologist, LAB = Laboratory, PHARM = Pharmacy) * refers to implicit patient consent.



4 Results

We started with a narrative scenario (see Appendix C for the full text) that describes a sequence of events in the common environment of paper-based forms and records dealt with by means of personal contacts through visits, telephone/fax and (e)mail. Each identified event or step was given an identifier to which we will refer in the remainder of this article. In the following we will refer to the EMR system of a specific health care provider by his acronym, followed by EMR (e.g. the EMR of the dermatologist is referred to as DERM-EMR). Figure 13 shows a sequence diagram for this scenario.

4.1 ISSUES ARISING FROM THE SCENARIO

Analysis of the scenario resulted in a set of questions. For reference, the entire list of questions is included in Appendix D. The questions were then grouped and rephrased to reveal the underlying issues:

- *Authorized access*: how to implement authorization across organizational boundaries? (Questions 1, 2, 6, 7, 8 and 11)
- *Confidentiality*: how can it be determined that no confidentiality breach has occurred when a copy of the information resides in another system? (Question 10)

- *Patient consent*: should patient consent be given implicitly or explicitly? Can patient restrictions be overridden by emergency procedures? Can patient restrictions be used to implement "deletion" of information? Can existing patient consent be extended to a new receiving party? Can a patient grant additional ad hoc access permissions to their EHR to meet unanticipated healthcare situations? (Questions 3, 4, 12, 13 and 14)
- *Relevancy*: how to define what information is relevant and when? (Questions 3, 9, 16, 17 and 24)
- *Ownership of information*: who is the owner of information and what are the implications for exchanging information across organizational boundaries? (Questions 6a, 19 and 23)
- *Infrastructure*: what are the implications of sending correction notifications on data that is passed on to third parties? (Questions 15, 20, 22 and 25)
- *Audit log*: what are the function and content of an audit log and the consequences for implementation in EHR systems? (Question 21)
- *Archiving*: what are the implications of data retention policies on the EHR content? (Question 26)

4.2 DISCUSSIONS OF THE ISSUES

4.2.1 Authorized access

Related questions:

1. *How should a patient be identified reliably across organizations?*
2. *How should health professionals be identified reliably across organizations? How should organizations be reliably identified?*
6. *How should the PSYCH-EMR system define authorization of the GP to access information in the system?*
7. *Should all systems have authorization information for all possible users (i.e. persons requesting information)?*
8. *Should all systems provide similar access for all possible users (i.e. a GP has access to the same kind of information in all systems)?*
11. *In case the information from the PSYCH-EMR is stored in the GP-EMR system and matches a future query from an external system (e.g. the query from the DERM-EMR), should the information be passed on if patient consent permits or should external information always be excluded from a result set?*

EMR systems contain sensitive information. It is a common requirement that access to such systems needs to be restricted to authorized users.
Reliable authorized access to patient information has three components: reliable patient identification, proper authentication of the health care provider and correct authorization of that provider.

4.2.1.1 *Identification*

Authorized access starts with a correct identification of both patient and healthcare provider. Within a single organization, it is sufficient to assign a unique code to a patient. Beyond the organizational borders, reliable patient identification across systems can be done in two ways:

- The identifications in various systems are retained and a Master Patient Index (MPI), a central service, is used to resolve collation [19].
- A national health ID (NHID) is issued by a governmental institution. In general these NHIDs are issued at birth, but can also be issued when someone starts to use the services of a health care system, for example in case of immigration. All identifications in the various systems should be replaced by or mapped to the national health ID.

The healthcare provider needs to be identified as well. Recently the Dutch government has issued a national health provider ID (UZI) [20] which can be issued to health care providers as well as to health care organizations. This ID is intended for unique identification of health care providers in cross-organizational information exchange.

The main problem of the MPI is the correlation: the reliable mapping of various patient identifications to a single physical person. Studies have shown that it is not possible to fully automate this process. [21]
The NHID is often described as a solution to the correlation problem, since it would require a one-time investment at the organizational level. It would also render an MPI superfluous. Many, mainly European countries, therefore regard the NHID as the more favorable approach.
Three issues arise in an environment with an NHID:

- The implications of identity theft and privacy breaches are larger.
- When patients cross national borders for health care, they will either receive a second (country-specific) ID or the local systems have to be capable of using the ID provided by the home country of the patient. Since the latter case is less realistic than the former on a pan-European scale in the present climate (due to difficulties in achieving consumer acceptance of such large scale identity sharing), an MPI to connect foreign and home IDs is still necessary.

- Patients without a national ID still need treatment and therefore a temporary ID, which needs to be correlated or replaced later with his or her national ID.

These issues would warrant the choice for an MPI, even in an environment where an NHID exists. The NHID can be seen as a solution to reduce the correlation problem, not a solution to bypass the MPI.

National IDs for healthcare professionals enable the reliable handling of situations of information exchange and are necessary to make it possible for patients to grant or deny access to their information for a specific healthcare professional (see § 4.2.3). A unique ID at organization level would be insufficient to handle this situation.
These IDs should already be issued to students attending medical school; from the moment there is a need to access patient information, which might be much earlier than required by current policies.

Health care organizations should also be uniquely identified. This can be used to mark the original creation location of data. Care should be taken that the organizational identification is not used to construct a unique ID for the healthcare professionals, since it would falsely imply a static relationship between the organization and the professional.

Finally, unique identification should be extended to all objects, ranging from information items and system components via documents to devices and systems that are used in obtaining and maintaining health care information. This is a standard IT approach as illustrated by the IMEI number (International Mobile Equipment Identity) in cell phones and the DOI (document object identifier) used to identify scientific publications. Relevant standards of the GS1 organization[22] could be used.

4.2.1.2 *Authentication*

A centralized authentication service is common practice in an organization, since it provides various levels of efficiency. Extrapolation to a distributed environment however, raised the question of how to authenticate external access. This can be done in two ways:

- The user is known beforehand, i.e. his identification credentials are registered in the system he wants to access. This implies a separate procedure to register the credentials.
- The system relies on credentials that are issued by other systems, either from

a different organization or from a general governmental body. The Dutch BIG registry, which registers all health care professionals [23], is an example of the latter.

Central registries, such as the BIG registry, can be used to grant or revoke credentials for a specific professional based on his professional behavior. Legal procedures for updating and querying the status of the credentials should be created.

4.2.1.3 *Authorization*

After being granted access to the system, various levels of authorization are still necessary. Authorization can be defined as access to data or as access to functionality of the system. In the first definition, access levels vary from access to the entire health record of a patient to fine-grained access definitions at the level of medical concepts. In the latter case, users are authorized to perform specific functions (e.g. order medication, Admission Discharge Transfer). Authorization to functionality implies authorization for access to data that is necessary to use the functionality, which in turn requires semantic interoperability. For the discussion at hand, the exact distinction is irrelevant.

Each user of a system may have several roles, usually a combination of static or structural roles, based on their position in the organization, and functional roles, based on their participation in the care process for a specific patient. Operations on resources (whether tasks or information objects) are defined as permissions. The basic relationship between roles and operations underpins Role Based Access Control (RBAC). The set of operations that a certain role can perform on specific resources can be defined and extended through a policy [24], [25], [26]. A policy can also include constraints, both static and temporal, to avoid conflicting accumulation of permissions and to comply with regulations of various kinds. The use of policies extends the capability of RBAC to handle richer specifications of privileges. ISO TS 22600 (Privilege Management and Access Control) defines a formal architectural approach to represent such policies, and for the services that need to be implemented to support their use within a distributed computing environment. [27, 28] Several sources use the term profile, but fail to sufficiently define it [9, 24, 29]. We define a profile as the set of constraints on the permissions assigned to a user, usually represented through their (structural and functional) roles and as a corresponding set of policies.

Several issues can be discerned:

- The higher the number of resources (whether tasks or information), the more complex the RBAC management will become. Evered et al. [30] have shown that RBAC implementation for a simple EHR system (nursing home with some

30 residents, 5 different structural roles and 4 sets of information) is already complex. Lovis et al describe an implementation of an RBAC model in the Geneva University Hospitals [31]. Although multiple hospitals are involved, they act as one organization using the same roles. Lovis advocates distributed RBAC management, close to the location of data access, because the users involved frequently change roles. Profiles grant access to coarse-grained objects such as system components and applications. They are further constrained by medical service and location, i.e. a nurse can only access patient information from a computer in her own ward. Lovis et al have implemented an emergency override procedure that allows annotated access to otherwise restricted information. This procedure is also used to fine-tune the implementation, because audit logs have shown that the emergency procedure is used very often.

- Users often claim to have very specific tasks, which cannot be modeled by generic roles or policies [32]. Adherence to the RBAC model will result in a myriad of roles unless an organizational policy to conform to generic roles is defined and rigorously implemented. However, care should be taken that the result does not interfere with good clinical care [33].
- Variations in access policies are quite common. Bakker [32] has described variations between departments, where physicians with similar roles have different authorizations. This can easily be extrapolated to organizations.

Inter-organizational connections add additional issues:

- Inter-organizational connections also require definitions of RBAC roles and profiles that do not exist in the internal organization, e.g. a hospital granting GPs access to the hospital's EHR should also implement a GP role and a profile for GPs, although there are no GPs working inside the hospital. The situation can be even more complex in practice, as individuals may have roles that combine policies from multiple issuing authorities, for example by an employing healthcare organization and a national health system and a professional body. Ensuring that none of these policies contain conflicting rules/constraints may at times be challenging.
- The RBAC model also implies that communicating parties agree on the definition and meaning of the role and profiles, i.e. a healthcare worker with a specific role has access to similar kinds of information in various systems. Extending this to all systems requires the existence of a universally applicable model of role definitions that is adopted by all health organizations. The American standard ASTM E1986-98 [34] has defined such a list, based on the American definition of roles. ISO DTS 21298 defines a similar set of structural and functional roles and refers to the International Labour Organisation

[35]. Since both do not give definitions or descriptions of each role and its policies, the only comparison can be done on the name of the role, yet from the standards it is not clear that e.g. the term "nurse" and the policies for the role of nurse are identical or at least comparable between ASTM and ISO.

Allowing patients to restrict access to their information reveals several discrepancies:

- If resources in RBAC models are defined as tasks that align to business processes, the patient cannot provide proper access restrictions, as he is concerned with the information that might be accessed by several tasks.
- If resources in RBAC models are defined as information objects, i.e. 'medication', access rules are defined for these objects, where patients would like to define (grant or deny) access to specific instances of these objects, e.g. allow access to 'medication' while restricting access to 'medication' instances from the PSYCH-EMR.
- If a patient can grant or deny access to a single person, rather than a role, a universally unique user ID, rather than an ID issued by the organization, is necessary to implement this restriction, since this person might not be a member of the organization at the time the access restriction is issued. It would also imply that to consistently deny a person access to the entire virtual EHR of the patient, this restriction has to be communicated to all other systems, preferably by a distributed component. In other words: role-based access needs to be extended with access rules for single individuals, thus increasing complexity.

In summary, ISO TS 22600 defines a structural and architectural approach to representing all of the information that may be required in policies in a standardized form. However, a complementary challenge is how to enumerate the classes of user, classes of EHR and kinds of constraint rule *interoperability* such that specific instances of policies can be recognized and applied consistently in diverse settings. There must be semantic interoperability regarding authorization roles and their respective profiles to allow consistent authorized access of information to external parties. The CEN 13606-4 standard recognizes this. A true global consensus on RBAC roles and policies seems very unrealistic, since it would affect the local organization of roles and might even be incompatible with national definitions. However, since the broad role concepts of "doctor" and "physiotherapist" are virtually identical throughout the world, we think that this consensus can reach a workable level. Also, all roles and policies should be implemented in all systems, even if they are not applied to internal users, to allow properly defined authorized access to external users.

As we have stated before, RBAC management is already very complex on a small

scale and will quickly increase in complexity on a regional, national and global scale.
Therefore, we believe that more research needs to be done to properly define an authorization service that has the flexibility of RBAC but can overcome its flaws. This research should take the proposed environment as a starting point to consider the described issues. ISO TS 22600 has made considerable steps in that direction.
To allow patients full control over the access restrictions to their virtual EHR, the discrepancies between tasks vs. roles and objects vs. instances need to be addressed. A universally unique ID for healthcare users would also be necessary.

4.2.2 Confidentiality

Related questions:

10 If PSYCH-EMR information is stored in the GP-EMR system, how can the PSYCH-EMR system be informed of possible confidentiality breaches?

As stated in the requirements confidentiality requires that proof can be given that the information has not been made available or disclosed to unauthorized entities, whether persons or systems. This can be implemented in two ways: either information is tagged with metadata about the confidentiality status or confidentiality is enforced by access rules.
Either way, if information is shared with authorized persons (e.g. the discharge information in STEP 2), there is no possibility for the sending system to verify whether this confidentiality has been breached (e.g. when the information is passed on to third parties). Since it can be argued that it is now the responsibility of the receiving party, there should be a legal and/or technical framework that regulates this issue. This implies that access profiles regarding the information should be added to the confidentiality tags and functionality in the receiving system should be present to execute these profiles. This requires that both systems use compatible profiles. When RBAC primarily focuses on permissions on tasks, this should be solved in a different way.
Using access rules to enforce confidentiality relies on the audit logs to verify that confidentiality has not been breached. This option also cannot verify if confidentiality breaches occur at the receiving end.

4.2.3 Patient consent

Related questions:

3 Should all information (always) be available unless restricted by the patient or

legislation or should information only be available if legislation and patient consent permit? At what level of granularity should the patient give his consent? Are there distinctions in types of situations? Should delegation be allowed, e.g. only if the GP thinks it is relevant?

4 *Following from the previous question: Is it legally acceptable to ask for patient consent for access by unknown health-related external parties? I.e. any doctor or any nurse versus a specific, named person.*

12 *What happens with this type of restriction if the patient moves from one GP to another?*

13 *When an emergency override is necessary, what access restrictions have still to be obeyed?*

14 *Can the right of the patient to delete information from his EHR be sufficiently covered by a total access restriction?*

Patients can allow or deny sharing their information with other healthcare workers. Consent must be either implied or explicitly given before the act of sharing.

4.2.3.1 *Implicit or explicit*

Patients either implicitly or explicitly consent to sharing information. This covers not only the exchange of information, but also the access to information by others than the original creators. Implicit consent assumes patients to have consented unless they specifically state otherwise. This is also referred to as opt-out. Explicit consent, or opt-in, is the reverse, where access to the information is prohibited unless the patient has given consent [36].

There is much debate about consent management. Studies have proven [37, 38] that the opt-out model results in simplified management and a higher number of included patients. This is also the proposal of the English NHS [37-43].

Advocates of the opt-in model state that it is the only guarantee to preserve patient privacy, which is considered to be more important than simplified management. The Royal College of General Practitioners [44] advises an opt-in approach.

Dutch legislation is in line with the EU directives, that the patient's agreement with the transfer of his health information should be laid down in an explicit consent [45]. However, in an attempt to keep the work process manageable but still compliant with legislation, the Dutch WGBO[12] law [17] defines a list of standard situations that assume implicit patient consent as well as a list of situations in which patient consent should be actively requested. Some examples: implicit consent is assumed for granting access to other healthcare providers participating in the care process

12 This law defines the rights of patients and healthcare professionals regarding health care.

and for access in emergency situations; explicit consent is necessary for granting access for a new episode of care or for research purposes.
Either way, managing patient consent within an organization is already complex. It adds an extra complexity level to the already complex RBAC model.
Implementation of patient consent is necessary for the purpose of automatic determination whether sharing of information is legally allowed. In Coiera et al. this is described as the gatekeeper system. [36] The system is able to remind the user that consent is denied or should be actively requested.
The focus of patient consent, as it is presently being tackled in e-Health programs, is on access to information of a particular kind and for a particular purpose, while the RBAC model can focus on access to information or to operations on information. A discrepancy is possible between the set of information the patient consents to share and the set of information the user has access to. An example to illustrate this: a patient is treated in a clinic for sexually transmitted diseases (STD) and has given consent to access his medication records created in the STD-clinic, but only to the physicians in the STD-clinic. His treating physician in a general hospital, who is authorized according to his RBAC profile, to review the medication records, should be denied access to the medication records in the STD-clinic.

4.2.3.2 *Internal versus external*

In Step 2 of the scenario, the patient restricts (partial) sharing of his psychiatric information to his GP only. This not only supports the argument that all available roles should be defined in a system, even if they are not fulfilled by actual organization members, but also raises the issue of the extent of patient consent. If the consent is fully specified for the internal situation of an organization, there might still be open issues when external (health-related) parties access the information, as shown by the STD-example above.
On occasions, the patient's consent to information disclosure needs to be captured explicitly, with a signature, for example to disclose health data to an employer or insurer. In such situations, paper systems are still usually used, but in future, a form of digitally signed policy is more likely.
A more complex challenge for both the STD and the psychiatric scenarios above is the implicit assumption that the whole record created within one care domain can be protected from access by all other clinical domains. Whilst this could probably be done in theory, the challenges for patient safety have not yet been adequately understood: will medical errors arise if vital information from those ring-fenced care settings is not provided to others treating the patient? A common proposed approach to this is to include within the overall policy framework some kind of override mechanism.

Due to the fact that an EMR system is currently designed for usage within an organization, no definitions are available about emergency overrides by external parties, e.g. an ER-physician from another hospital.
A very probable situation where the patient consents to the requesting party during an encounter, rather than beforehand, has also not been covered. This could result in a delay in therapy when the patient has to contact the originating party, unless the current treating physician is allowed to use an emergency override procedure to access the information.
The fourth part of the 13606 standard defines emergency procedures to override access restrictions [9]. The standard does not specify in detail which access restrictions should be overridden.
The EU directive Articles 7 and 8 (EU 95/46/EC) [45] demand that the patient's agreement with the transfer of his health information should be laid down in an explicit consent [46], unless he is incapable of doing so in situations where it is of vital interest to share the data with a health professional. EU directives also imply that the organization sharing the information is responsible and should verify that the requesting party is indeed a qualified health care professional. This implies that information is shared between natural persons. It is unclear what the legal implications are when patients consent to generic roles, which can be fulfilled by hitherto unknown persons.
In the context of the Dutch new ICT infrastructure all health care providers (both persons and organizations) will be uniquely identified through a so-called UZI-number [20]. Organizations can use this number to access external information of a patient. However, when the UZI-number of the organization is used the external system cannot properly identify the actual person that requests the information and is therefore unable to respect the patient's wishes. Even when transferring along with the information the responsibility to honor the security profiles and patient consent to the requesting organization, full identification of the requesting person and his role is necessary.

4.2.4 Relevancy

Related questions:

3 *Should all information (always) be available unless restricted by the patient or legislation or should information only be available if legislation and patient consent permit? At what level of granularity should the patient give his consent? Are there distinctions in types of situations? Should delegation be allowed, e.g. only if the GP thinks it is relevant?*

9 *Dutch legislation [WGBO] only allows access to "need to know" information.*

Should systems be capable of deducing what information a GP needs to know, and if so, how can this be achieved?

16 *Should the dermatologist be able to request all available information or only what is relevant?*

17 *Should the dermatologist be able to access all available information of the patient (present in various systems) at any point in time or only when relevant?*

24 *Should the dermatologist be allowed to access the patient information after the discharge letter is created?*

Dutch legislation [17] states that a healthcare worker who uses an EHR should not access information unless it is relevant for the management of the patient. Similar legislation can be found in other countries. The General Medical Council in the UK states in its regulations for Good Medical Practice [47] that physicians should not disclose information about their patients except in specific cases.
In the past, the owner of the data could raise questions about the relevancy of a request, but in an automated exchange, this is not possible. New Dutch legislation is likely to enforce GPs to make their data available through the Dutch infrastructure defined by NICTIZ (Dutch Institute for ICT in Health care) [48]. So the burden of determining relevancy is now completely with the requestor of the information.
This requires a clear definition of both the information that is allowed to be shared and the situations in which sharing is allowed, i.e. an unambiguous definition of 'relevancy' that pertains to both content (what is relevant) and time (when is it relevant). Yet, relevancy is a very ambiguous concept that is also highly dependent on the context and therefore very hard to define in such a way that implementation in a software system is feasible. This is recognized by Lovis et al. [31]. They have solved the issue by requiring a well-defined care relationship between user and patient and by relaxing the time constraints. This approach is less suitable in a cross-organizational situation since the requested system cannot reliably verify a suitable relationship between the intended external recipient and the patient.

Conversely, relevancy prevents data overload: allowing a user to view all available data can result in losing the overview and the capability to make an informed decision. Striving for the display of only relevant information becomes a pragmatic goal.

The ideals of relevancy include not only the right to know (if a suitable care or consented relationship exists with the patient) but also the need to know (granting access only to the health record information that is needed by the requestor to perform relevant tasks). However, determining the latter is almost impossible in advance for any individual patient and any individual access scenario (i.e. when data

are committed to an EHR or when policies are being defined). In our opinion relevancy of information can rarely be defined a priori and can only be implemented in a few well-defined, unambiguous situations. It is more pragmatic to allow access when in doubt and verify possible abuse afterwards (e.g. through the audit log).
Prevention of data overload could be implemented by a layered structure that shows an increasingly detailed view of the available information. On the top level, only the different types of information are available. A second level shows summaries, while the lowest level shows detailed data. This allows users to drill down to the most interesting, i.e. most relevant, data while noticing the availability of other information types.

4.2.5 Ownership of information

Related questions:

6a How can the GP trust the information?

19 If the information is assumed to remain with the owner, who is the owner? The person ordering the information or the person generating the information? Or both?

23 Should the GP be able to differentiate between the various owners of the information included in the message?

The current practice of storing both locally and externally created information in the same system requires the identification of the owner and the origin of the information. The origin of the information is often defined as the location where the information is created, i.e. entered into the system.

In general, the owner is defined as the creator of the information. Establishing the owner of the information is necessary for several purposes: the owner is responsible for the availability of the information [46], for the accuracy of the information[13] [17] and for protection against unauthorized access [49]. The owner is also held responsible in legal disputes. Note that the 'owner' can refer to the person responsible for the information or to the organization storing the information.

Ownership is closely linked to origin. Information that originates from an external system has a different owner and should be processed differently. Blobel has argued that data should be traceable to its origin and that data should be kept only

13 Initial accuracy is the creator's responsibility, who is also the owner at the time. After transfer of the ownership, the new owner might be held responsible for the accuracy.

at the origin [50] to avoid redundancy and to maintain integrity.
Clear definition of the ownership of data is necessary to resolve the question whether the lab is the owner of the lab results or the physician ordering them, because it is directly relevant to the issues mentioned above. Only the lab can send corrected test results (maintain accuracy), but the dermatologist is responsible for acting on them, and for including the correct lab results in his discharge letter.
Definition of ownership is also important to resolve the situation where a patient chooses a different GP and all of his information has to be transferred. The new GP has not created the information, but on using it (and thereby trusting it), he is also responsible for assessing the accuracy of that information.

There is some ambiguity in the above definitions. The Dutch WGBO law states that the creator of the information is the owner of the information [17]. NICTIZ however, differentiates between the author and the "manager" of the data [51]. The manager has not created the data but incorporates the information in his/her own system and assumes responsibility. An example to clarify this is the Dutch situation where many regions have a central General Practice locum service (HAP) where GPs are on duty in the evening or weekend. When a patient is treated at this HAP, a summary message is sent to the regular GP who incorporates the information in his local system. From that moment on, the source of the data is the regular GP's system. The GP is also the manager, while the locum remains the creator. This distinction solves the problem of transferring ownership. However, it also implies that data is not strictly stored at the place of creation (i.e. the system of the locum).
This led us to define the concepts of owner and origin more precisely and make a distinction between the creation and the management of the information.
The discussion above shows that the definition of 'owner' is very ambiguous. Legislation in different European countries also differs on defining ownership of the data in an EHR, which further increases the ambiguity. We therefore propose to avoid using the term "owner" and want to introduce the following terms:

- "**creator**" for the person generating the data and/or entering the data into the system;
- "**author**" for the person or entity responsible for the content of the information, and
- "**manager**" for the person or entity responsible for the management, provision and protection of the information.

Similarly, the source is the location of the information that is managed by the manager, while the origin is the location where the information is created. We assume that in most cases, both persons and locations respectively are the same.

TABLE 6 Definitions of ownership

	Definition
Source	Location where the data are stored
Origin	Location where the data are created
Manager	Person/Entity responsible for the data (provide, protect)
Author	Person/Entity responsible for the content of the information
Creator	Person generating the data and entering it in the system

Correct definition of manager, author and source are directly related to data integrity and non-repudiation (both receiving and sending) requirements of the EHR. It is possible to define business and legal policies for situations where the manager and/or author of information changes (e.g. when a patient changes to a different GP) or where information from several authors is exchanged with third parties (e.g. the discharge letter from the dermatologist to the GP that includes test results). In the latter case, there should be clear definitions on how the external information should be incorporated to avoid conflicting authorization policies and outdated information.

4.2.6 Infrastructure

Related questions:

15 How should the query be propagated to all available systems in the area?

20 How should the dermatologist be informed of the corrected information?

22 Should the dermatologist be able to view both the current (i.e. new) and the old (i.e. those sent before the recalibration) values?

25 What action should be taken if the test results were updated after the discharge letter was sent?

Information in a system is subject to changes caused by corrections such as the lab test rerun described in the scenario (STEP 11). Audit logs capture the correction of data, while version management ensures that each value (i.e. version) of the data is clearly identified.

Exchanging information with external systems requires a notification mechanism to inform (external) recipients of the correction.

4.2.6.1 *Version management*

Version management is necessary to uniquely identify a specific value of a specific concept at a specific point in time, for example in recreating a specific situation during an audit.

This implies that the version management of the system is capable of retrieving

not only the most recent version, but also the version that was previously seen by the user. This in turn implies that queries to external systems should include parameters that can be used by the system where the data resides to retrieve the correct version.

Version management in distributed systems requires the use of unique IDs for each version of all relevant objects, whether persons, devices or data structures and for a common identifier for all versions of the same object. This ID is separate of, but connected to the object's ID, which should also be available in each version.

4.2.6.2 *Notifications*

- When information is updated, all relevant parties should be notified, including external parties, which implies that the system providing the information registers the requesting parties in such a way that this information can be accessed and used.
- In general, notifications are used at a lower OSI level to handle asynchronous processes such as transactions. In this section, we discuss notifications at an application level where they are used to inform interested parties of updates. Notification is not only used to secure data integrity, but also to inform health care professionals that they have (possibly) based their decision on possibly invalid data.
- If we look at the scenario where the dermatologist writes a discharge letter based on the test results available and the lab sends corrected test results, two situations can be distinguished:
- Only decisions based on external information are described in a document that is sent to a third party. In this case the corrected information is not included, but might result in different decisions and therefore an updated document.
- External information is incorporated in a document that is sent to a third party (e.g. test results are included in the discharge letter sent by the dermatologist to the GP). In this case not only the dermatologist should be notified, but also the GP, regardless of a possible update of the discharge letter with a different decision.
- Both cases imply that the sending system should have a registration of previously requested information and the identification of the requesters.
- This also requires the registration of all external information that is incorporated in information that is sent to third parties. If the corrected information matches the external information, the user should be informed and an updated version of the including information should be sent to the third party. A question arises if the user should be informed of the corrected information if

he has not seen the previous version, since he has not acted upon the invalidated information. Whatever the outcome of this question is, it always requires a registration of which user has retrieved what information. Another approach is to only send notifications of corrected information when the information has been queried before. This implies that an EHR system not only registers the senders of the queries, but also tags the information with properties to determine if previous versions of the updated information have been queried.

- In all cases, an interoperable notification framework is necessary.
- It is not clear where to locate this notification registry. Although this information is also logged in the audit log (see § 4.2.7), we believe it is against the purpose of the audit log to be actively used by the EMR system. This would result in a very convoluted link between the EMR system and the audit log. Setting up such a registry in the EMR results in duplication of information, which is also not a desirable situation.
- There is also concern that the automatic notification of every EMR data update to every previous potential recipient is a logistic nightmare, and may result in unwanted disclosures (if, for example, previous recipients are no longer caring for the patient).

4.2.7 Audit

Related question:

21 Should the system be able to log the fact that the dermatologist has actually seen the (updated) information?

The audit log should document all actions performed on the information and the users performing those actions, to enable the recreation of the state of the past[14]. There are various reasons for this functionality:

- It can be used for security purposes, i.e. to monitor the access and possible misuse of the system, preferably in real-time.
- It can be used for review purposes; EN 13606-4 [9] specifies how patients might be provided access to audit log information to review access to their EHR. This implies that the log information should be converted to something that is understandable by the patients. Security policies should be applied for the auditors to avoid security breaches. The Dutch infrastructure of NICTIZ will have this functionality implemented.

14 Note that clinical audit, which we define as the auditing of the medical information for the purpose of improving health care, is beyond the scope of this article.

- It can be used in legal disputes to verify claims about what information was available and whether it was accessed. This can also be used for non-repudiation issues.
- It can be used to complement access control to update and fine tune RBAC policies as well as to verify the relevancy of the accessed information.

The logging should not be restricted to the information handled, but should include all events and state changes. This results in a huge volume of data that should be stored for (possible) future reference, while it should also be possible to process the log in real-time to detect intrusion or unusual behavior. The use of an emergency over-ride mechanism to access otherwise prohibited health record information is an example of this.

Due to the distributed nature of information retrieval in the context of the scenario of this article, information and audit logs are also distributed. In order to track the full details of an action (e.g. a query and its results) many systems and subsequently many audit logs are involved. This trace across systems requires interoperable audit logs to allow unambiguous processing. To faithfully trace a specific event across systems all events need to be uniquely identified through a generic identifier, either by the same identifier throughout all systems or by mapping the external identifier to an internal identifier. Furthermore, a synchronized timestamp is necessary to order the events and functionality is required to retrieve a specific version to allow faithful recreation of a past situation.

Although the interoperability is recognized by the EN 13606-4 standard [9], no interoperable audit logs currently exist. The IETF RFC 3881 [52] provides a draft specification. ISO and CEN are currently working on a standard for a common framework for interoperable audit logs based on this RFC.

The requirement to maintain an audit log can be implemented in an audit service. Audit logs should be separated from the EMR system, with a clear definition of the connection between the two to maintain data integrity on both components. This allows the use of the audit log as described above, without placing a load on the EMR system itself. In addition, different authorization profiles apply to the audit log, those of audit managers, not of healthcare professionals. Logging actual data in the audit log should be avoided, because of the security issue. Otherwise restricted information would then become available to non-healthcare professionals (e.g. audit managers). By logging a reference to the information, rather than the actual information, in a separate system both requirements can be met. The audit logs relating to emergency access to the EHR, which often overrides usual access policies, should be reviewed extra carefully to ensure that such privileges are not abused. A separate log event with details of the emergency action could be cre-

ated in a separate, highly restricted service, with a minimal reference in the regular log. The latter would be sufficient for routine audit procedures, while the former can only be accessed after proper authorization. Emergency overrides should be investigated to determine the necessity. This requires recording of the situation that triggered the override, information that is often not recorded.
In some countries, patients have the right to have data removed from their health record. This would imply that the audit log contains references to non-existing data. Provisions are needed to cope with this situation. Data removal by patients also poses problems when trying to recreate past situations, since the data is not available any more and should be marked as such.
Accessing the audit log itself should also be logged for the same security reasons, although with the theoretical risk of creating audit logs ad infinitum.
Storage requirements of an audit log can be considerable which might lead to policies to keep logs only for a defined time span. Since the cost of storage devices is continually decreasing, the only reason for a policy of purging audit logs should be legal requirements.
Use of the information in the audit log might have medico-legal implications. Based on the logs it might be deduced that certain information was accessed without imminent need. It might even be deduced that information was not accessed when access could be expected. Although it is obvious that unauthorized access should be investigated, what are the implications of the absence of access?
This discussion shows the importance of a clear definition of the purpose and content of an audit log as well as a clear model of the connection between EMR system and audit log.
Such a definition would dictate the kinds of information that an audit log should contain, and the ways it should be monitored.

4.2.8 Archiving

Related question:

26 What should be done with the data after legal data retention time has passed?

Archiving is the mechanism of moving data out of the active system into locations, which are less immediately available, usually for storage logistics and performance reasons. Wherever possible, archived data should be technology-independent so that future users do not have to depend on obsolete technology from the past.
Dutch legislation [17] defines that medical data should be kept for at least 15 years. Longer periods are allowed if it is essential to the health of the patient or even to

the health of his relatives (e.g. hereditary diseases). Many countries require longer data retention periods for liability issues.
This implies that systems should be able to retain this information for at least the period of the legal data retention. For such a long time, the issue of the ownership is even more important, but it also complicates queries, because older information is usually less relevant in everyday treatment.
Not only the data, but also the authorization profiles pertaining to that data should be preserved. Roles and profiles will evolve over time and reliably answering auditing questions can only be done when the exact context of the question is recreated, including the authorization profiles at that time. This in turn implies that version management on roles and profiles is necessary.
Records spanning a lifetime with multiple care providers both in parallel and in succession and restricted by both organizational and patient restrictions might lead to data that should be retained due to its age and/or importance but is not available due to its authorization profiles.
In paper-based records older information is often manually summarized, e.g. after each episode of care, reducing the size and superfluous details in the data to be interpreted in future episodes of care. This does not imply that the detailed data is removed. The electronic record could also provide summaries while retaining the detailed data. There is no standard yet that deals with the action of summarizing electronic data, either manually or automatically. There are also no guidelines on how to assess, which information is still relevant for the patient's situation after the retention time, who should make this decision, and on what criteria this decision should be based.

Dutch legislation allows patients to ask for removal of data from their record. This not only overrules the data retention policy, but also contradicts the ISO 18308 requirement that data should not be deleted. Data retention policies and lifelong health records are also contradicting since the latter requires all information available during the lifetime of the patient, while the former requires destruction of information older than a certain age.
Finally, documentation obligations of organizations (e.g. for health statistics or liability requirements) conflict with the patient's right to remove data. One approach could be to allow the patient to hide direct access to information using a mechanism outside the regular access control mechanism, e.g. by using encryption with the patient's key (thus requiring the patient's cooperation for decryption), whilst permitting it to be included in population queries that are needed for statutory reporting.

Data retention policies and lifelong health records require long-term storage. This means that data should still be accessible long after the creation date. This implies storage in future-proof formats and/or keeping old software and hardware components around to be able to access it. It puts considerable strain on the various implementations.

Shabo [12, 13] recognizes these problems and describes a solution where health records are maintained at health record banks equivalent to financial banks. He suggested that the health record banks should be funded indirectly through the health insurance plans of the patients opening an account. More recently, a number of large commercial players have launched personal health record systems. Although this solution moves the burden of long time storage to a central health bank or central commercial server, the problem of future accessibility will remain.

The question arises to which extend should past situations be recreatable. This period could be much shorter than the retention policies. In special cases, manual procedures are probably the most cost-effective.

4.3 FUNCTIONALITY AND IMPLEMENTATION OF THE CENTRAL SERVICE

The central service as depicted in Figure 12 plays a vital role in the described environment. Literature shows two approaches for implementation of such service, which we will describe here:

- The medical information is stored in a central system: a repository.
- Pointers to the medical information are stored in a central system: an index service.

The repository can vary from a set of (pointers to) attested documents about specific aspects of the medical history of patients [53] to a full record of all available information [12, 13] and anything in between. A centralized index service or registry is currently implemented by the Dutch NICTIZ [54]. They are compared and contrasted here in relation to the issues described before.

4.3.1 An IvEHR repository

If the central service were implemented as a repository, it could contain documents covering only information that is considered of relevance for properly dealing with future health issues, like discharge letters and medication lists. Its advantage is the single point of retrieval of attested documents. No discussion on the quality and no extended searches of connected systems for information are necessary. The authorization issues still apply since only authorized persons should have access to the documents. The confidentiality issue can be avoided by adding to the repository only documents that have the same confidentiality tags.

Patient consent can be handled easily because it is known beforehand what will be added to the repository and what user roles are allowed to review the information. Relevancy is handled by a strict definition of the content of the documents. No corrections are necessary (i.e. documents are aggregated/summarized from underlying information).
The documents are summaries or aggregations of information on different levels of granularity, with the actual information still residing in the various connected systems. The documents contain patient identifiable information and therefore become a privacy risk. The documents should be tagged with IDs from the authors and source, for proper definition of the ownership and responsibility of the data. Since the repository does not necessarily hold all available information, additional information either cannot be retrieved or should be retrieved through an indexing service. Connected systems are still responsible for the information and contain a duplicate of the information that is contributed to the repository.
An alternative is the PING architecture described by Simons et al [55]. This is a centralized, cross-organizational repository of documents where the patient can define fine-grained control in delegating access to portions of his EHR.

Shabo goes one step further and claims that lifelong patient-centered EHRs can only exist when storage and maintenance is transferred to a health record bank (similar to a financial bank). The health record bank can transfer a copy of the EHR (all or in part) to a specified health care provider after patient consent. During treatment, the health care provider can add information to the EHR, which is transferred back to the health record bank after discharge. Intermittent updates of the original EHR in the health record bank are necessary when treatment takes a considerable amount of time, e.g. in a chronic condition.
The health record bank takes care of the archiving problems and the authorization, although the authorization issues remain. All available information on the patient is in his EHR account, so an indexing service for additional information is not necessary. It cannot be avoided that patients open several accounts to hide sensitive information. There is no possibility for the health record bank to verify confidentiality breaches when the EHR is transferred to a health care provider, unless all audit information pertaining to the specific record is also added to the account. It is unclear if patients can consent to sharing only part of the information or can restrict access to specific persons or roles. RBAC models are likely to differ between care providers so it is not clear which models (care provider or patient) will take precedence. It is also not clear how emergency overrides should happen when a patient has not consented to transfer information to a specific care provider and/or the EHR is not (yet) transferred to the care provider. When a patient receives treatment

at different locations, careful synchronization of information is necessary to avoid conflicting treatments due to lack of information.

In both lvEHR and Health Bank repository-types, ownership is a problem: as stated before according to Dutch law, the person entering the information in the health record is the owner of the information and is responsible for its accuracy and completeness. Other countries have different legislation, making a single approach across countries very difficult.
The repository of attested documents relies on the connected systems to keep the respective documents up-to-date and complete. The same goes for the health record bank.

Authorized access in hospitals is often linked to a specific relationship with the patient. It will be difficult for the health record bank to verify such a relationship.
EHR systems storing all incoming information (as most current systems do) will be superfluous and may have no legal status in the environment of health record banks.

4.3.2 Index service

An alternative implementation of the central service would be as an index service. An index service does not store the information on a patient, but merely index pointers to the actual source of the information.

Various indexes are possible:

- A *passive* index, which acts as a telephone directory: the index returns locations of the information which are in turn queried for the actual information;
- An *active* index, which acts like a broker: the query is addressed to the index service. The index service in turn propagates the query only to systems that have registered the information in the index that matches the query. Finally, the combined result set is returned to the query originator. The Act Reference Registry (ARR) of HL7 as implemented by NICTIZ is an example of an active index service.

To avoid returning unallowed information a broker system should be able to either pass on the user ID and RBAC profiles that are attached to the queries, or be able to match the profiles to the RBAC information that is added to the registered information.
Registered index pointers should only be visible to users with appropriate authorization. Querying would be most efficient when only the systems are addressed

that will return information for that user. This implies that a broker system is capable of executing RBAC policies. However, only the originating system is responsible for filtering unauthorized access and updating the local RBAC profiles. To execute the RBAC policy by the broker would imply that the entire set of policies of all registered systems should be duplicated in the broker system. This seems to be an unacceptable situation.
A separate complicating factor is the granularity of the information index. Coarse-grained indices can be fewer and so more manageable, but may lack sufficient details for RBAC policies and/or patient restrictions.
Legislation might deny storing privileged information in a central location. This effectively reduces the options to a passive or active index service.

5 Discussion

5.1 DISCUSSION OF THE APPROACH

Our method for identifying the relevant issues and their elaboration is similar to the first two steps of the methodology of the HL7 Message Development Framework (MDF) [56] for defining messages.
In very global steps, the HL7 approach starts from a realistic scenario or storyboard through the definition of use case models and interaction models, using UML modeling tools, to specific messages. A scenario is a more informal description of the events, which is better suited for capturing the time-sequence of events than the more formal use case models. It also hides details of the Actors involved in the interaction.

Following the HL7 MDF approach allowed us to focus on a specific scenario, which simplified the definition of the questions. Only one scenario was defined, but it represents a generic, privacy sensitive situation. Each institution or department can be replaced by a different equivalent where sensitive or general information is generated, without changing the overall issues.
By focusing on real life processes and asking simple questions, we believe the essence of an issue is much better clarified than defining an abstract goal that should be met. An example: RBAC is considered to be a good solution to define access rules for users and its implementation would meet the goal of secure systems. However, a question like "how could a dermatologist request information from a GP system" shows that implementing RBAC in both systems is not sufficient to solve the problem raised by this question.
Defining questions is an endless process; there will always be more questions to

TABLE 7 Mapping of issues to EHR requirements

Issue	EHR requirement
Authorized access	Authentication/Authorization/Patient access
Ownership	Integrity (of data)/Non-repudiation
Ownership	Author responsibility
Confidentiality	Confidentiality
Patient consent	Consent
Relevancy	
Audit	Audit/Non-repudiation
Version management	Audit
Notifications	
Archiving	Archiving

ask. By grouping the questions in issues, which are then mapped onto the topics of various standards, it can be argued that the question set is sufficiently complete when all issues cover all topics.

Table 7 shows that the topics of data integrity and non-repudiation have not been mapped to a separate issue but depend on proper identification of the author/manager of a record contribution and appropriate audit logs. Patient access is also not covered separately, because it would make the scenario even longer and more complex, and because a patient can be seen as a user with a special role and profile and therefore not different from other users such as health care providers.

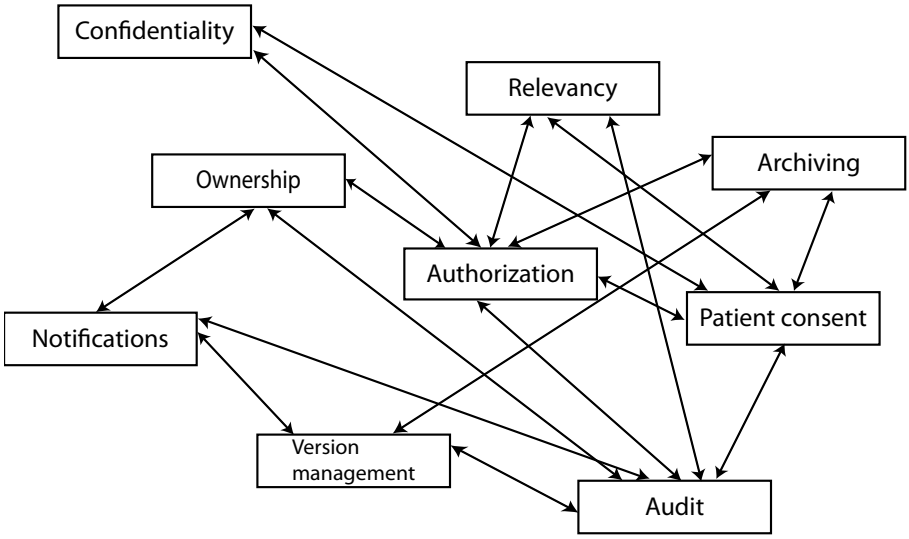
The only mention of relevancy in the ISO definition is the requirement that only data relevant to the care process should be recorded. It is not mentioned as a separate requirement. Version management also has no corresponding requirement but the discussion shows that it is a vital pre-requisite for auditing. The issue raised by notifications has not been covered by any ISO requirement although it touches on data integrity.

Unraveling the issues gives insight into the problems that can arise. Literature review has shown various studies into these problems and possible solutions. However, most reviews focus either on a single issue or on a limited environment. We have found that the issues influence each other and a solution for one issue can cause additional problems for another issue.

Figure 14 gives an overview of the relations between the issues. We see that all issues are interrelated, but that authorization, patient consent and audit play a pivotal role.

We used an approach where we discussed and studied the issues in the order presented in this paper. Information and possible solutions from an earlier issue were taken into account into later issues. This sequence was repeated several times,

FIGURE 14 Relations between issues



while taking into account all the information that was revealed in earlier rounds. When no significant changes in the information occurred the issue was regarded as "closed".

5.2 DISCUSSION OF THE RESULTS

The lvEHR can only become reality when there is a truly secured network of EMR systems that contain information about the patient. Thus, in theory each patient has a different network, i.e. comprised of different EMR systems, which is not fixed over time since it can grow when the patient visits other healthcare providers/organizations. As described before, there are some issues to be solved before these networks will be in operation.

RBAC as described in this article is considered more suitable for simpler environments. More complex environments such as health would benefit from policy-driven RBAC, as described in the Privilege Management and Access Control (PMAC) framework. This ISO specification [27, 28] extends RBAC by defining a framework to represent and manage computable policy agreements between parties to exchange and use information. The policy agreements specify which information to exchange and under which security-related circumstances. Although these policy agreements simplify the problem by focusing each policy on one type of exchange under a specific set of well-defined circumstances between parties, the level of overall complexity rises with the number of agreements since these each need to

be maintained and updated to reflect current changes and also need to be in accordance with each other's other policies to avoid conflicting policy agreements.
We believe that PMAC will certainly solve most problems mentioned in this article at a very restricted level of cooperation between a few parties. Expanding the cooperation to include more information and more parties at a greater regional and national scale will require a lot of resources to keep all policies in line.
Since policies agreements are defined up front and go through a standardized definition process, there is no possibility of securely exchanging new types of information that have not yet been agreed on.
We think the current paradigm shift from paper-based to the current EMR systems with their local storage of all incoming information is not sufficient to transparently incorporate a virtual lifelong patient record that spans multiple independent healthcare organizations. The shift should be extended to move towards an infrastructure that resembles the environment described in this article with a focus on interoperability and sharing of information in a privacy enhanced way.
True interoperability of systems pertaining to an lvEHR can only be guaranteed if the privacy issues mentioned before are solved. It also entails not only semantic interoperability of the information, but also of the authorization roles and profiles as well as all other aspects pertaining to networked EMR systems.
We have focused on the security issues in an environment of interconnected EMRs/EHRs. Currently the notion of a Personal Health Record (PHR) is elaborated. The PHR is seen as a solution to guard against identity theft and unauthorized access since the only way that data can travel is from an EMR into the PHR. With the patient in control of the PHR, data filtering is performed by the patient (some data will be included, other data not). Access control is also executed by the patient, who can decide during a consultation whether to share existing data or not with the consulted physician. We haven't analyzed the security issues related to the PHR in detail. We see that besides the security issues, other aspects like the impact of unshared data on patient care etc. play an important role in the further development of the PHR. Such issues are outside the scope of this manuscript.
A promising technology for implementation of the described environment would be peer-to-peer (P2P) networking [57], where all the participating systems share the computing and storage resources to fulfill the demand. This would result in a truly transparent EHR system. However, since P2P also entails replication of (parts) of data on several nodes, it raises security issues along the lines, described in this article. More research into that area is necessary before it can be useful.

6 Conclusion

In the age-old tradition of medicine, healthcare workers have developed a way of communication that follows the lines of a conversation. Information is specifically requested from defined other persons. Typically, even today the preferred means of communication are still telephone, fax and email. Current information retrieval as defined by HL7 and 13606 is still very much based on this practice.
The currently omnipresent search engines on the Internet that provide access to vast amounts of information have shifted the paradigm from a one-to-one exchange to a one-to-many exchange and from a tendency to store everything locally to a tendency to search and retrieve whenever the need arises. It is time to implement that paradigm in the EMR systems and the lifelong virtual patient record.

Before the described virtual lifelong patient record can become reality, more clarity has to be provided on the following:

- The implementation of the authorization model.
- The implementation of patient restrictions and patient consent in general.
- Legal and computational frameworks that protect confidentiality.
- A definition of relevancy that is legally accepted and machine interpretable.
- A future-proof infrastructure that supports communication between systems in different organizations given the above restrictions.
- Archiving information for future use given the situation where people and systems are changed frequently in the course of the life of the patient.

Technical solutions need not and will not replace all personal communication. The EHR as a shared source of information for professional discussion can pragmatically solve some of the issues (such as the relevancy issue) discussed in this article.

7 References

1 Grimson W, Berry D, Grimson J, Stephens G, Felton E, Given P, et al. Federated healthcare record server–the Synapses paradigm. Int J Med Inform. 1998 Oct-Dec;52(1-3):3-27.
2 Grimson J, Grimson W, Berry D, Stephens G, Felton E, Kalra D, et al. A CORBA-based integration of distributed electronic healthcare records using the synapses approach. IEEE Trans Inf Technol Biomed. 1998 Sep;2(3):124-38.
3 Grimson W, Jung B, van Mulligen EM, van Ginneken AM, Pardon S, Sottile PA. Extensions to the HISA standard–the SynEx computing environment. Methods Inf Med. 2002;41(5):401-10.
4 Harmonisation for the secuRity of web technologies and aPplications. 2000 [cited 2008/02/18]; Available from: http://www.telecom.ntua.gr/~HARP/HARP/INSIDE/Inside.htm
5 HL7. [cited 2008/02/15]; Available from: http://www.hl7.org

6 Health Informatics – Electronic health record communication - Part 1: Reference Model: CEN/TC251. Report No.: EN13606-1:2007.
7 Health Informatics – Electronic health record communication - Part 2: Archetypes: CEN/TC251. Report No.: EN13606-1:2007.
8 Health Informatics – Electronic health record communication - Part 3: Reference archetypes and term lists. EN13606-3:2008: CEN/TC251.
9 Health Informatics – Electronic health record communication - Part 4: Security requirements and distribution rules: CEN/TC251. Report No.: EN13606-4:2007.
10 Health Informatics – Electronic health record communication - Part 5: Messages for exchange: CEN/TC251; 2007. Report No.: prEN13606-5:2007:E.
11 openEHR. [cited 2008/02/15]; Available from: http://www.openehr.org
12 Shabo A. A global socio-economic-medico-legal model for the sustainability of longitudinal electronic health records. Part 1. Methods Inf Med. 2006;45(3):240-5.
13 Shabo A. A global socio-economic-medico-legal model for the sustainability of longitudinal electronic health records. Part 2. Methods Inf Med. 2006;45(5):498-505.
14 Jaspers MW, Knaup P, Schmidt D. The computerized patient record: where do we stand? Methods Inf Med. 2006;45 Suppl 1:29-39.
15 ANSI. ISO/TS 18308 Health Informatics - Requirements for an Electronic Health Record Architecture: ISO; 2003.
16 ANSI. ISO/TR 20514 Health informatics - Electronic health record - Definition, scope and context: ISO; 2005.
17 Wet Geneeskundige Behandel Overeenkomst (WGBO). [cited 2008/02/18]; Available from: http://www.hulpgids.nl/wetten/wgbo.htm
18 Anderson RJ. Security in Clinical Information Systems. 1996 [cited 2008/02/18]; Available from: http://www.cl.cam.ac.uk/users/rja14/policy11/policy11.html
19 Person Identification Service Specification. 2001 [cited 2008/02/18]; Available from: http://www.omg.org/technology/documents/formal/person_identification_service.htm
20 Dutch Unique Healthcare Provider Identification Register (UZI-register). [cited 2008/02/18]; Available from: http://www.uziregister.nl/
21 Bittle MJ, Charache P, Wassilchalk DM. Registration-associated patient misidentification in an academic medical center: causes and corrections. Jt Comm J Qual Patient Saf. 2007 Jan;33(1):25-33.
22 GS1. [cited 2008/02/18]; Available from: http://www.gs1.org/
23 Registratie en informatie beroepsbeoefenaren in de zorg. [cited 2008/02/18]; Available from: http://www.ribiz.nl/
24 Role Based Access Control. [cited 2008/02/18]; Available from: http://www.csrc.nist.gov/rbac/
25 Role Based Access Control Feb 2004. Report No.: ANSI INCITS 359-2004.
26 Blobel B, Nordberg R, Davis JM, Pharow P. Modelling privilege management and access control. Int J Med Inform. 2006 Aug;75(8):597-623.
27 ISO/TS 22600-1 Health informatics - Privilege management and access control - Part 1: Overview and policy management: ISO; 2006.
28 ISO/TS 22600-2 Health informatics - Privilege management and access control - Part 2: Formal models; 2006.
29 eXtensible Access Control Markup Language (XACML). [cited 2008/02/18]; Available from: http://docs.oasis-open.org/xacml/cd-xacml-rbac-profile-01.pdf
30 Evered M, Bögeholz S. A case study in access control requirements for a Health Information System. Proceedings of the second workshop on Australasian information security, Data Mining and Web Intelligence, and Software Internationalisation - Volume 32. Dunedin, New Zealand: Australian Computer Society, Inc.; 2004.
31 Lovis C, Spahni S, Cassoni N, Geissbuhler A. Comprehensive management of the access to the electronic patient record: towards trans-institutional networks. Int J Med Inform. 2007 May-Jun;76(5-6):466-70.
32 Bakker AR. The evolution of Health Information Systems, security in practice and open issues. Stud Health Technol Inform. 2003;96:15-20.
33 Han YY, Carcillo JA, Venkataraman ST, Clark RS, Watson RS, Nguyen TC, et al. Unexpected increased

mortality after implementation of a commercially sold computerized physician order entry system. Pediatrics. 2005 Dec;116(6):1506-12.
34 Standard Guide for Information Access Priviliges to Health Information: ASTM; 2005. Report No.: E1986-98.
35 ISO DTS 21298 Functional and Structural roles: ISO.
36 Coiera E, Clarke R. e-Consent: The design and implementation of consumer consent mechanisms in an electronic environment. J Am Med Inform Assoc. 2004 Mar-Apr;11(2):129-40.
37 Adams T, Budden M, Hoare C, Sanderson H. Lessons from the central Hampshire electronic health record pilot project: issues of data protection and consent. Bmj. 2004 Apr 10;328(7444):871-4.
38 Hewison J, Haines A. Overcoming barriers to recruitment in health research. Bmj. 2006 Aug 5;333(7562):300-2.
39 Norheim OF. Soft paternalism and the ethics of shared electronic patient records. Bmj. 2006 Jul 1;333(7557):2-3.
40 Cundy PR, Hassey A. To opt in or opt out of electronic patient records? Isle of Wight and Scottish projects are not opt out schemes. Bmj. 2006 Jul 15;333(7559):146.
41 Watson N. Patients should have to opt out of national electronic care records: FOR. Bmj. 2006 Jul 1;333(7557):39-40.
42 Wilkinson J. Patients should have to opt out of national electronic care records: what's all the fuss about? Bmj. 2006 Jul 1;333(7557):42-3.
43 Junghans C, Feder G, Hemingway H, Timmis A, Jones M. Recruiting patients to medical research: double blind randomised trial of "opt-in" versus "opt-out" strategies. Bmj. 2005 Oct 22;331(7522):940.
44 BMA statement on Connecting for Health. 2006 [cited 2008/02/18]; Available from: http://www.bma.org.uk/ap.nsf/Content/cfhstatement
45 Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data. [cited 2008/02/18]; Available from: http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31995L0046:EN:HTML
46 van der Haak M, Wolff AC, Brandner R, Drings P, Wannenmacher M, Wetter T. Data security and protection in cross-institutional electronic patient records. Int J Med Inform. 2003 Jul;70(2-3):117-30.
47 Good Medical Practice. 2006 [cited 2008/02/18]; Available from: http://www.gmc-uk.org/guidance/good_medical_practice/index.asp
48 Nationaal ICT Instituut in de Zorg. [cited 2008/02/18]; Available from: http://www.nictiz.nl/
49 Nouwt S. Beveiliging van het EPD. Rapportage van het juridisch laboratorium. Den Haag: ZonMW/ICZ; 2002. p. 65.
50 Blobel B. Advanced and secure architectural EHR approaches. Int J Med Inform. 2006 Mar-Apr;75(3-4):185-90.
51 Implementatiehandleiding HL7v3 Zorg Informatie Makelaar. [cited 2008/02/18]; Available from: http://www.nictiz.nl/uploaded/FILES/AORTA%20release%20augustus%202006/hl7_zim%20v2.5.pdf
52 RFC 3881 Security Audit and Access Accountability Message XML Data Definitions for Healthcare Applications. 2004 [cited 2008/02/18]; Available from: http://tools.ietf.org/html/rfc3881
53 Machbarkeitsstudie betreffend Einfuhrung der elektronischen Gesundheitsakte (ELGA) im osterreichischen Gesundheitswesen. 2006 [cited 2008/02/18];
Available from: http://www.arge-elga.at/fileadmin/user_upload/uploads/download_Papers/Arge_Papers/Machbarkeitsstudie_ELGA_Endbericht_21112006.pdf
54 Architectuurontwerp basisinfrastructuur in de zorg, versie 4.2. [cited 2008/02/18]; Available from: http://www.nictiz.nl/uploaded/FILES/AORTA%20release%20augustus%202006/Architectuur%20AORTA%20versie%204.2.pdf
55 Simons WW, Mandl KD, Kohane IS. The PING personally controlled electronic medical record system: technical architecture. J Am Med Inform Assoc. 2005 Jan-Feb;12(1):47-54.
56 HL7 Message Development Framework. [cited 2008/02/18]; Available from: http://www.hl7.org/library/mdf99/mdf99.pdf
57 Peer-To-Peer Networking. 2008 [cited 2008/02/15]; Available from: http://en.wikipedia.org/wiki/Peer-to-peer

Chapter 8

Generic screen representations for future-proof systems, is it possible?

There is more to a GUI than meets the eye

(to be continued)

Published as Helma van der Linden, Tony Austin, Jan Talmon, Generic screen representations for future-proof systems, is it possible? There is more to a GUI than meets the eye, Comput Methods Programs Biomed. Vol 95, 2009, p 213-26. Epub 2009 Apr 15.

Generic screen representations for future-proof systems, is it possible?

1 Introduction

Interoperability is considered a key property of the generation of electronic health records (EHR) systems to come. It will allow EHRs to communicate and to interpret the data received.

The two major standardisation approaches in this respect are HL7 v3[15] [1], where the architecture of the messages between systems are standardised, and CEN/TC251 13606 combined with archetypes[16] [2, 3] which standardises the record itself.

As advocated by *open*EHR [4], true future-proof electronic health record systems will be able to accommodate new medical concepts without the need for large adjustments to the system [5]. This is possible because a Reference Model is implemented in the information system, which need not change, and "archetypes" then express structured medical concepts as constraints to and combinations of these classes. A consequence of this approach is that patient information in structures that have not been previously encountered can still be satisfactorily interpreted by the system.

Although future-proof systems are capable of comprehending previously unknown (ad hoc) information structures since they still adhere to the underlying Reference Model, there is currently no method for the optimal display of such data. A clinical application receiving data for which it has no predefined presentation format will have to resort to a low level generic representation of the received data that may be difficult to understand by the user, for example, a family tree might be presented in the form of a nested XML structure because the application has not been configured for a graphical display of this information. In order to present this kind of information in the future the application will need to be modified, and continually so for each newly encountered data structure. This is clearly not a scalable approach, given that clinical data structures do evolve and EHR systems in the future will receive patient data from other widely distributed and heterogeneous EHR systems using different data structures. Throughout this article we will use the non-trivial yet easy to understand example of a blood pressure concept. In a local system, it may suffice to define a blood pressure by the systolic and diastolic values. However, when the blood pressure is communicated to others, contextual infor-

15 In this article, HL7 will refer to the new v3 standard in its latest form.

16 In this article, we refer to the CEN 13606/archetypes as archetypes for brevity.

mation such as the cuff size and the position of the patient are required to make a proper interpretation. Over time, definitions of concepts may change and become more complex and new concepts may be defined in particular areas of medicine. Systems that are part of a larger network need to be able to deal with these evolving definitions, preferably without a (major) effort in the redesign of the software. This article discusses an approach to solve these problems by developing a systematic way of representing and implementing presentational knowledge.

1.1 BACKGROUND

Interoperability is the ability of two or more systems or components to exchange information and to use the information that has been exchanged [6]. Both HL7 and CEN 13606 aim to achieve interoperability both at the data structure level and at the domain model level. At the lowest level, medical concepts are described using predefined data structures. This ensures that the information exchanged is complete (it contains all relevant and required data and metadata) and can be parsed, stored and subsequently retrieved. At the higher domain level, additional metadata are used to avoid ambiguity in understanding. For example, in textual values these might include the code and a reference to the coding scheme.

The PropeR project studied the architecture of a generic electronic health record (EHR) system. During this study, a prototype implementation was built and tested [7, 8]. The system was configured for use as multidisciplinary EHR by several therapists in primary care in the context of rehabilitation of stroke patients in their home environment.

PropeR showed that the successful use of received information is not simply a matter of communications between systems, but also between the receiving system and its users. For example, even if the GP's system is capable of storing and subsequently retrieving a family tree, it is very difficult for the GP to correctly interpret the information if there is no suitable screen representation.

1.2 THE CEN 13606 ARCHETYPE APPROACH

The two-model approach of CEN 13606 consists of a Reference Model with predefined classes that can be implemented in software and an Archetype Model that defines a UML model that constrains and combines the classes from the Reference Model to express medical concepts in standardized structures. This approach separates the software development from the medical knowledge implementation and permits clinical users to define structures describing their clinical specialities without needing to understand how those structures will be exchanged or committed to persistent storage.

An archetype or archetype definition is a description of the structure of a medical concept, as it would be documented in an EHR system. For example, a blood pressure archetype describes two physical qualities, one called systolic, the other called diastolic, each having a unit of measurement, along with optional metadata for the position of the patient, the cuff size used and the time of the observation. These values are represented in a containing class labeled "Blood Pressure". Since archetypes support object-oriented principles, generic archetypes can be further subspecialized. This allows the definition of a generic concept such as a "laboratory test" to be specialized into a "blood sugar test" or a "complete blood count" (CBC). Actual archetypes derived from the UML model are also referred to as archetype instances. An actual observation, for example a specific blood pressure measurement in a particular patient, is referred to as an EHR instance.

1.3 MOTIVATION FOR THE PROJECT

In the context of the PropeR project [7, 8] a web-based EHR system was built using a simplified version of CEN 13606 archetypes. We focused on the implementation of a domain-agnostic system based on these archetypes, with generic screen representations. The feasibility of generating a GUI based on archetypes [9] was studied in a second phase project. Both projects revealed that a generic GUI would result in a suboptimal display where the structure of the information is used as the only basis to derive the presentation, rather than displaying the information in a form familiar to the user. It was found not to be possible to support the definition of an optimal display format while keeping the GUI generic.

The two-model approach, which is the basis for the archetypes, has proven to be useful in separating software development from knowledge implementation. This inspired the authors to apply the same approach to the GUI domain, in an attempt to enable the generation of good quality and useful presentations of EHR data without requiring that each data structure (archetype) be known in advance during the system design.

2 Methods

The objective of the project reported in this paper was to develop a new framework for presentation-level interoperability. However, since it was expected that the proof-of-concept implementation would elicit further requirements and require iterative development cycles, we focused on reusing already available tools for implementation. Two existing candidate presentation frameworks were tested for the proof-of-concept implementation:

1 The Cocoon Forms Framework, based on XForms and suitable for data entry as well as data display [10].
2 The XML User Interface Language (XUL), the Mozilla Foundation framework for developing GUIs [11].

These frameworks were chosen to take advantage of existing experience from earlier projects [9, 12] and to allow for rapid implementation. It was also hoped that the choice of industry-standard tools would ensure that the focus remained on the feasibility of the new presentation layer and not be diverted towards other implementation issues relating to the development of a new technology. However, it was recognized that neither framework had been tested for this purpose previously and that it might later become necessary to develop a proprietary (albeit open) alternative. This might need either to be an amalgam of the two above, or entirely new from scratch.

In all cases, the implementations were integrated in the Apache Cocoon Web application to actually deliver the application pages. A set of various EHR instances was presented via both implementations.

The authors also tested the generic nature of the approach by mapping the framework to an HL7 structure representing a blood pressure, to assess the work that would be required for this.

2.1 PRESENTATION-LEVEL INTEROPERABILITY

Irrespective of the purpose of an element of displayed information, it is likely that the presentation of information via an application will have used three types of knowledge:

- Knowledge of the best way to display the information (content-related).
- Knowledge of the way a user is accustomed to view information (localization).
- Knowledge of the device that is used to display the information (device-related).

These different types of knowledge are often mixed and hard-coded into the GUI of the client application. This not only makes it difficult to display information from a domain different to the one for which it was designed, but it also necessitates duplication of presentation knowledge within the application to accommodate different display devices.

2.1.1 Content-related presentation knowledge

Content-related presentation knowledge relates presentation behavior to the constituents of the information structure; for example, numbers might be displayed differently from text. Such simple knowledge, however, is not sufficient. In medicine, more complex information structures exist. For example, a common visual

form for blood pressures is of two numbers separated by a slash (Systolic/Diastolic). An upside-down tree might be the best visual form of a genealogical history.

2.1.2 Localized presentation knowledge

Visual information may be subject to local customisation, from the local language and date format to the preferred units (e.g. mg/dl vs. µmol/l) and for the use of different coding schemes. These may be established countrywide, or within an individual institution. There are also personal preference differences such as learned behaviour or different cognition preferences (e.g. pictorial, textual).

2.1.3 Device-related presentation knowledge

There is now a large range of devices capable of sending and retrieving information on behalf of a user. They include not only desktop computers and laptops, but tablet PCs, PDAs and smartphones. The aspiration for smaller devices today is towards achieving the same browsing experience as that provided by the more full-featured devices like desktops. However, this is not universally realised yet and the presentation toolkits must ideally retain the flexibility to modify content based on the rendering device.

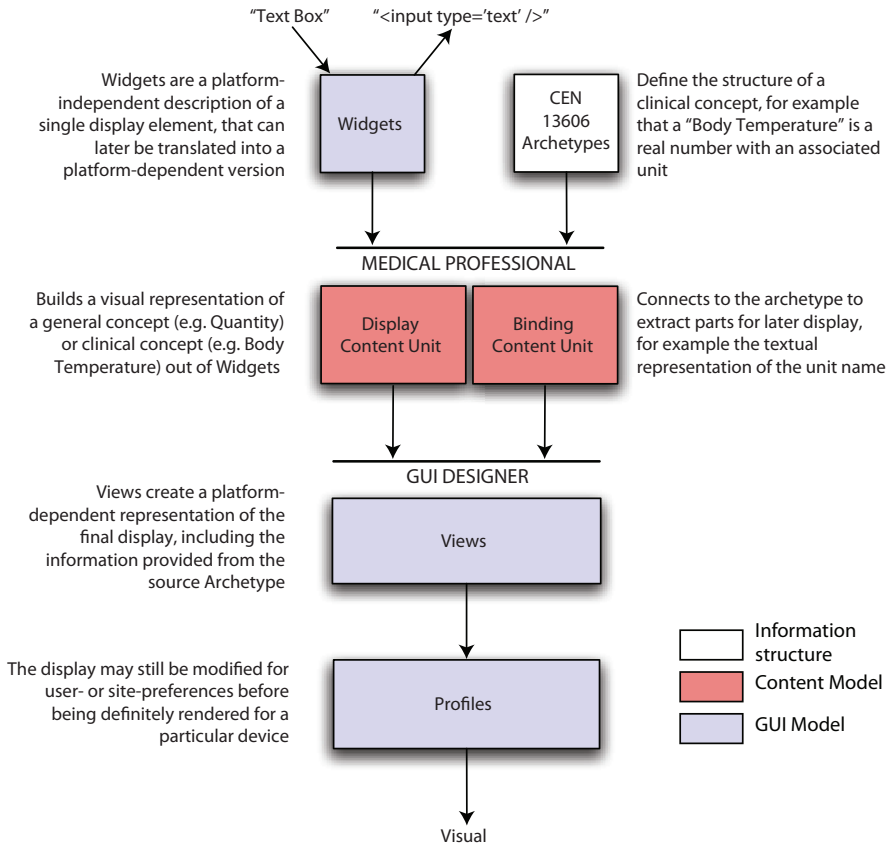
2.2 A TWO-MODEL APPROACH TO GENERIC GUI GENERATION

From the PropeRWeb application the authors identified that including screen presentation knowledge in the archetype definitions introduces dependencies that make content modification difficult [12]. Not only does it introduce two different kinds of knowledge (medical domain knowledge and presentation knowledge) into a single model, it also adds complexity to the display of the same information in different ways according to context.

Moreover, adding specific display information to the archetype would duplicate the effort of definition. For example, if numerical information from different medical concepts could be displayed in a similar way, the definition for that display would still have to be added to each archetype that represents numeric data. This approach would also increase the effort to maintain archetype definitions if each one had to be updated whenever a display definition was added or modified.

The authors distinguish two models: a display-oriented model (the GUI model) that defines Widgets as screen presentation units; and a domain-oriented model (the Content model) that defines Content Units, which create meaningful presentations using Widgets. The first model is the realm of the GUI designer, while domain experts use the second model. If this is compared to the archetype approach, the GUI model resembles the Reference Model, while the Content Model resembles the Archetype Model. During development, a strict separation of do-

FIGURE 15 Diagram showing the various model components and their relation.



mains was maintained echoing those archetype characteristics.

An overview of how a visual is generated using the two models is given in Figure 15. The process is based on the concept of pipelines. Various units handle binding and widget selection. Localized presentation knowledge is defined in the final profile.

Given the premise that future-proof systems are also capable of receiving information from other domains, it is necessary that these systems contain domain-agnostic presentational functionality. All units follow object-oriented design principles in which a specific unit inherits characteristics from a more generic unit. This improves consistency and flexibility, and implies that multiple units can be defined for any archetype to broaden the range of possible displays available.

2.2.1 GUI model

The GUI model consists of:

- Widgets, the building blocks of a GUI.

- Views, the definition of a screen.
- Profiles, which tailor the presentation to the local environment.

Widgets are commonly known in software development as the elements that make up the windows of an application, such as text boxes and labels. A Widget in this approach is a platform-independent display unit that contains presentation knowledge for a single data type. These widgets can be mapped to classes in the underlying Reference Model. Two types of widgets exist: data-oriented widgets such as "text", "image" and "number", and group-oriented widgets such as "list", "table" and "graph". The definition of these widgets is typically a one-time investment since both the Reference Model (and therefore the widget set), is stable over time.

Widgets are converted to specific (device-dependent and/or platform-dependent) versions in the underlying system by using Views. A View is focused on presentation of the content and therefore part of the GUI designers' remit. Views provide the transformation from platform-independence to platform-dependency.

In a View, the screen/window is divided in parts, each with a different purpose. Each part is bound to a data source. This can be as simple as a static image for a logo or a predefined tree for navigation bars, up to more complex sources such as the demographics of the current patient for the header part and the content units for the content part. An application is therefore not expected to create the "screen chrome" itself, but rather to devolve the responsibility for that to the View.

Profiles implement localized presentation knowledge. Profiles manage the conversion of information to match user expectations to avoid interpretation errors.

A profile contains preferences at various levels that modify the presentation of the information. There are three levels:

- *System level.* This level contains generic preferences that should always be applied, e.g. language, date format, and metric vs. imperial units.
- *Local level.* This level contains generic preferences that are organization or location specific and are more domain-related. These preferences include preferred units and preferred terminologies. This level updates preferences on a per-role basis.
- *User level.* This level contains user-specified preferences that can modify the preferred view for a certain type of content unit e.g. if the user prefers graphs to tables for the same content.

By combining these tiers, the GUI can be tailored to the user and organizational preferences.

2.2.2 Content model

The Content model describes the content-related presentation knowledge in

Content Units. A Content Unit consists of two parts: a Display Content Unit and a Binding Content Unit.

A Display Content Unit is a composition of one or more Widgets, combined with other display-oriented information to provide a platform- and device-independent description of the layout of a medical concept, which will have been described in an archetype. The authors suggest using a Content Unit Definition Language (CUDL), which is still to be defined formally, for this description. Display Content Units can be regarded as the display counterpart of archetypes. Like archetypes, they can include other Display Content Units.

Display Content Units can also include (references to) normal ranges for semantic interpretation of the value. For example, the value of a body mass index (BMI) can be color-coded based on the semantic interpretation (e.g. "normal" is green, "obese" is red).

The Binding Content Unit describes how the parts of the Display Content Units are connected to the associated parts in the archetyped EHR instance, through the Content Unit Binding Language (CUBL), which also needs to be defined formally.

Separation of the binding from the archetype allows for multiple content units that refer to a single archetype and provides the flexibility to display the same information in different ways. It should also be possible to map content units not only to archetypes, but also to other data structures.

Like archetypes, Content Units are stored in a repository in a format that is ready for use. The exact implementation could be anything from simple source code files to high performing databases. Since they are a platform-independent representation of the presentation of EHR data conforming to an archetype, they can be shared in the same way archetypes are sharable.

Any profile that specifies how a specific widget has to be displayed will be applied to the components of the content model that refer to such a widget. This mechanism makes it possible to apply local/system/user preferences also to data that was not seen before.

3 Results

A proof of concept Web application was built using the Apache Cocoon Web application framework [13]. This Web application can display EHR instances conforming to various archetypes based on the approach described above. The EHR instances are offered to the system as an XML structure.

FIGURE 16 Default display of a list of blood pressures

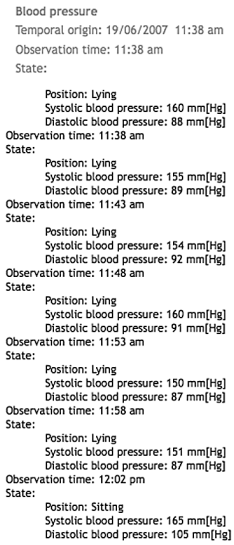
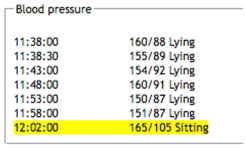


FIGURE 17 Same list of blood pressures, but using an optimized presentation, the last entry is highlighted to signal the change in position.



The Apache Cocoon Web application framework is a generic open source framework that is fundamentally based on the concept of separation of concerns. This is reflected in the component-based implementation and supports the corresponding pursuit of separation of knowledge types. It implements the pipeline concept as described above. Since Cocoon excels in processing XML, it is a good candidate to build a generic Web-based GUI that can be generated.

3.1 COCOON FORMS FRAMEWORK

The Cocoon Forms Framework, or CForms block [10], is a standard part of the Cocoon framework and provides widgets that can be used for both data display and for data entry. In its simplest form, a CForms widget is defined by three XML structures: the definition, the binding and the template. The definition is a description of the label and the data type. Optionally, validation rules can be added, such as checks on date ranges or lists of predefined values, but also more elaborate validations that can be added as functions in JavaScript or Java, because an API is available to access the widgets in these languages. The validation can be done, not only at the widget level, but also at the form level, i.e. across widgets.

The binding is based on XPath and connects the widget to its underlying information structure. The latter can be anything from XML to JavaBeans to SQL-results or even custom-built structures. In its simplest form, the binding merely provides the mapping between the CForms widget and the underlying data structure, but

in more complicated structures it also holds the definition of how to add or delete substructures (e.g. in a table where rows can be added and/or deleted).
Archetypes are hierarchical structures and support an XPath-like definition [14] to access substructures, which matches the binding used in CForms. The CForms binding file would be an implementation of the Binding Content Unit.
The template provides a platform-dependent description of the actual display of the widget, for example an HTML page where the label and the value are displayed as rows in a table or a text area used for the data entry of a string. Note that a (HTML) form is typically implemented in Cocoon as three different XML files, one for the definition of all the widgets, one for the binding of all the widgets and one for the template of the form, and its widgets. These files can be handcrafted or partially or fully automatically generated. The definition defines the aggregation of the CForms widgets and provides a simple ID to the template for further layout specification.
The template combined with the definition would be the implementation of the Display Content Unit.
Figures 16 and 17 show the same list of blood pressures. The first is generated by a generic template resulting in a long list, while the second is optimized by a Blood Pressure Content Unit.
CForms work well with other Cocoon blocks such as the internationalization block (i18n). Together they provide out-of-the box localization. The use of the i18n block in Cocoon implies that text that would require localization is tagged with i18n tags. Based on the provided locale, the i18n block replaces the tagged text with its localized counterpart. Other conversions such as a localized date format are also supported. The i18n block would be part of the implementation of the Profiles, which would reduce the amount of implementation work necessary.
The first version of the CForms implementation revealed that it is relatively easy to define content units based on CForms widgets for simple archetypes such as a body temperature. Using JavaScript it was possible to define validation rules across widgets and to write simple conversion rules for units. This allowed for on the fly calculation of the body mass index for example, and for quick conversions into a different unit.
The drawbacks of the CForms became evident when moving on to more complex data structures such as the blood pressure. CForms have no definition of a generic layout (e.g. horizontal orientation or grid), but rely on a set of available Extensible Stylesheet Language (XSL) stylesheets that provide a standard rendering in HTML. These stylesheets can be overruled, but are generally considered an integral part of the CForms block.
Therefore, in the template, it was not possible to define a generic display of the

blood pressure in the previously mentioned Systolic/Diastolic format without selecting a specific approach such as an HTML DIV or TABLE structure. This approach should be based on the overall design and device constraints and is therefore part of the view, not the content unit.

In the definition, it was very difficult to build a composite content unit based on a composition of widgets. To define a generic Quantity widget in CForms a composite CForms widget would be required. Standard CForms widgets consist of a label and a value for a specific data type. The label content is defined in the definition, rather than bound to an external information source. There is no provision for displaying a unit. The implementation of a Quantity widget in CForms would be a so-called class widget consisting of a common string widget for the label, a common widget with a numerical type and another common string widget for the units. The challenge lies in the mapping of multiple instances of similar values, e.g. if a list of blood pressure measurements is provided, they all map onto the same definition, but still need a mechanism that distinguishes them from each other.

The rudimentary implementation of validation rules using JavaScript showed that it is necessary to have a way to reference other archetype values as in the case of the BMI calculation where the values of the weight and the height archetypes are necessary. CForms widgets have IDs and, as stated before, validations can be defined on a form level, across widgets. The reference to the individual values in the BMI calculation would be easy to achieve in the CForms validation. However, a BMI calculation is part of the domain knowledge, not of the domain display knowledge. Therefore, it should not be part of the Content Unit, but of the archetype. This also raised the question where in general the validation, calculation and conversion methods should be stored.

Not all platforms support JavaScript. A platform-independent specification language is required for widespread adoption.

3.2 XML USER LANGUAGE (XUL)

XUL is a definition language created by the Mozilla Foundation and used for their products such as FireFox and Thunderbird. Apart from the usual data type oriented widgets it also provides widgets that are more generic such as grids and horizontally and vertically oriented boxes. This provides the necessary abstraction for the display and therefore solves the drawback of CForms' reliance on HTML for these types of widgets. XUL templates [15] provide a means to connect the XUL widgets to data sources. However, its implementation revealed that it lacks the flexible and easy binding of CForms and requires complicated definitions of templates, queries and data sources. Although this complexity provides a lot of flexibility, it makes it difficult for a non-programmer to define the necessary structures.

Other disadvantages are:

- XUL template data sources are subject to a 'same-domain' restriction. This effectively means that the XUL template and the referenced data sources should be part of the same URL domain. This not only prevents separation of the GUI model, the Content model and the data, but also prevents the generation of a screen based on multiple sources from different URL domains.
- The implementation of a XUL template and the data sources are part of the definition of the XUL widgets (e.g. 'datasource' is an attribute of a vbox, a vertically oriented box). This makes a separation of view and data binding impossible and prevents multiple binding definitions to a single view.

Schuler et al [9] also noted the immaturity of the language. Bowers also describes the immaturity of the XUL templates [16] in a web article, in which he points out that although the concept looks promising, the current implementation prevents development of an application beyond the simple examples provided.
Since XUL is only supported by Mozilla products, the final display requires a conversion to standard HTML or, if possible, the use of a Mozilla browser.

3.3 EXPERIMENTAL PROPRIETARY FRAMEWORK

Drawing from the experiences of both the CForms and the XUL implementation efforts described above, we defined a simple Display Content Unit Language in XML based on the XUL vocabulary, but added the XPATH binding of CForms. The necessary conversions were done using various Extensible Stylesheet Language (XSL) stylesheets. We focused on data display.
For each archetype, a matching Display Content Unit was implemented as a small XSL stylesheet. This XSL stylesheet puts the appropriate information in the label, value, and unit triple.
Without the functionality of the CForms validation rules, it was necessary to implement unit conversion rules in XSL. Although it was possible to achieve this, the attempt demonstrated that the available functionality in XSL is too limited and cumbersome to use. Unit conversions ideally require different mechanisms than simple XSL transformations.

4 Evaluation

4.1 ENHANCED EXAMPLE

The example below shows the necessary components and the process in the pipeline that provides a meaningful display of a blood pressure on a user's smartphone.

When the requesting system application receives an EHR instance containing a blood pressure, it looks up the reference of the archetype (1) in the EHR instance and finds a matching content unit (2) going from specific to generic in the archetype hierarchy. It then retrieves the appropriate archetype and content unit from their respective central repositories. A similar mapping process is executed to find an appropriate view (3) based on the targeted device and the references to the content unit. This mapping process is also performed from specific to generic. The view is requested from a local repository. Based on the user credentials the appropriate profile (4) is also retrieved from a local repository. All elements are combined to display the blood pressure onto the user's smartphone with his or her preferred language and formats.

More detailed descriptions of how the various presentational units could look are

FIGURE 18 Archetype definition of a blood pressure (source: the openEHR Foundation www.openehr.org)

```
archetype
        openEHR-EHR-OBSERVATION.blood_pressure.v1

concept
        [at0000]        -- Blood pressure measurement
description
        original_author = <
                ["name"] = <"Sam Heard">
                ["organisation"] = <"Ocean Informatics">
                ["date"] = <"22/03/2006">
                ["email"] = <"sam.heard@oceaninformatics.biz">
        >
        details = <
                ["en"] = <
                        language = <"en">
                        purpose = <"To record the systemic blood pressure of a person. The measurement records the systolic and the diastolic pres
                        use = <"All blood pressure measurements are recorded using this archetype. There is a rich state model for use with exerci
                        keywords = <"observations", "blood pressure", "measurement">
                        misuse = <"Not to be used for intravascular pressure.">
                >
        >
        lifecycle_state = <"AuthorDraft">

definition
        OBSERVATION[at0000] matches {   -- Blood pressure measurement
                data matches {
                        HISTORY[at0001] matches {       -- history
                                events cardinality matches {1..*; unordered} matches {
                                        EVENT[at0006] occurrences matches {0..*} matches {      -- any event
                                                data matches {
                                                        ITEM_LIST[at0003] matches {     -- blood pressure
                                                                items cardinality matches {0..*; ordered} matches {
                                                                        ELEMENT[at0004] occurrences matches {0..1} matches {    -- systolic
                                                                                value matches {
                                                                                        C_QUANTITY <
                                                                                                property = <"openehr::125">
                                                                                                list = <
                                                                                                        ["1"] = <
                                                                                                                units = <"mm[Hg]">
                                                                                                                magnitude = <|0.0..1000.0|>
                                                                                                        >
                                                                                                >
                                                                                        >
                                                                                }
                                                                        }
                                                                        ELEMENT[at0005] occurrences matches {0..1} matches {    -- diastolic
                                                                                value matches {
                                                                                        C_QUANTITY <
                                                                                                property = <"openehr::125">
                                                                                                list = <
                                                                                                        ["1"] = <
                                                                                                                units = <"mm[Hg]">
                                                                                                                magnitude = <|0.0..1000.0|>
                                                                                                        >
                                                                                                >
                                                                                        >
                                                                                }
                                                                        }
                                                                        ELEMENT[at0033] occurrences matches {0..1} matches {    -- Comment
                                                                                value matches {
                                                                                        TEXT matches {*}
                                                                                }
                                                                        }
                                                                }
                                                        }
                                                }
                                        }
                                        POINT_EVENT[at0002] occurrences matches {0..1} matches {        -- baseline reading
                                                offset matches {|PT0s|}
                                                data matches {
                                                        use_node ITEM_LIST /data[at0001]/events[at0006]/data[at0003]
                                                }
                                        }
```

FIGURE 19 Display Content Unit definition

```
<contentunit id='bloodpressure' archetype='openEHR-EHR-OBSERVATION.blood_pressure.v1'>
    <hbox>
        <label id='bp_label' widgetpath='datetime/value'/>
        <value id='systolic' widgetpath='number'/>
        <text>/</text>
        <value id='diastolic' widgetpath='number'/>
        <value id='units' widgetpath='text'/>
    </hbox>
</contentunit>
```

FIGURE 20 Binding of content unit to archetype definition. Note that the paths in the "value" definitions are relative to the path in the binding tag.

```
<contentbinding id='bloodpressure' path=
'//OBSERVATION/data/HISTORY[at0001]/events/EVENT/data/ITEM_LIST[at0003]/items'/>
    <value id='bp_label' path='/POINT_EVENT[at0002]/offset/value'/>
    <value id='systolic' path='/ELEMENT[at0005]/value/magnitude'/>
    <value id='diastolic' path='/ELEMENT[at0004]/value/magnitude'/>
    <value id='units' path='/ELEMENT[at0005]/value/units'/>
</contentbinding>
```


presented below. Simple XML notation was used for clarity. Paths are expressed in XPath notation.

4.1.1 Archetype (1)

An archetype definition of a blood pressure contains two ELEMENTs of type Quantity each consisting of a value (the actual measurement) and a unit (usually mmHg), marked with a name ("systolic" or "diastolic") (see Figure 18). Additional information such as cuff size, and patient position during measurement can also be described using this archetype.

As a clinician, one is typically only interested in displaying the date and time and the values of the blood pressure. The most common way of displaying this is in the familiar layout of: [time] S/D [unit], with S being the systolic blood pressure and D the diastolic blood pressure.

4.1.2 Widgets

As stated before, Content Units are based on Widgets, which in turn represent the data type classes of the Reference Model. Timestamps can be displayed by date-time widgets, numbers by number widgets and labels by text widgets.

4.1.3 Content units (2)

A specialized Blood Pressure Content Unit defines a blood pressure as a combination of a date-time widget with two number widgets for the values, separated by a slash and followed by a label to display the unit (see Figure 19). All this should be displayed in a horizontal layout. The content units also contain mappings that bind the widgets to the appropriate substructures of the archetype definition (see Figure 20).

FIGURE 21 Generic view definition and a similar definition for a smartphone

```
<view id='simplescreen' type='generic'>
 <vbox>
  <hbox>
   <part id='logo'/>
   <part id='header'/>
  </hbox>
  <hbox>
   <part id='navtree'/>
   <part id='content'/>
  </hbox>
 </vbox>
</view>
```

```
<view id='simplescreen' type='smartphone'>
 <table>
  <tr>
   <td><part id='logo'/></td>
   <td><part id='header'/></td>
  </tr>
  <tr>
   <td><part id='navtree'/></td>
   <td><part id='content'/></td>
  </tr>
 </table>
</view>
```


4.1.4 Views (3)

The GUI designer defines a generic view that defines a part holding a content unit for the patient identification information, a part containing a navigation menu and a part that serves as a placeholder for any type of content unit (see Figure 21, top definition). The specialized view in this smartphone example is derived by mapping the widgets to their smartphone equivalents, in Figure 21, bottom definition, shown as a simple HTML table. This view can also contain information on design such as font and font size choice, and colors.

4.1.5 Profiles (4)

A system-wide profile is created that defines the preferred language, the metric system, the date and time format and so on for the organization where it is implemented. If necessary, additional profiles could be created to accommodate the system to special users, e.g. visiting physicians from the USA.

4.2 MAPPING TO HL7 STRUCTURES

HL7 focuses primarily on message exchange and therefore considers this display problem to be part of the receiving system's domain. However, this approach is equally useful in that realm.

To test the generic nature of the approach the Content Unit of a blood pressure was mapped to the HL7 structure for a blood pressure. Since no HL7 tools were available to generate an XML version, the mapping was performed by hand using

FIGURE 22 Binding of content unit to HL7 RMIM-model of blood pressure

```
<contentbinding id='bloodpressure' path='//BloodPressure'>
    <value id='bp_label' path='@effectiveTime'/>
    <value id='systolic' path='/component1/SystolicPressure/@value'/>
    <value id='diastolic' path='/component1/DiastolicPressure/@value'/>
    <value id='units' path='/component1/SystolicPressure/@unit'/>
</contentbinding>
```

FIGURE 23 Binding of content unit to HL7 CDA Document Sample of blood pressure

```
<contentbinding id='bloodpressure' path='//entry/observation'>
    <value id='bp_label' path='/effectivetime/@value'/>
    <value id='systolic' path='/entryRelationship[observation/code/@code="271649006"]/value/@value'/>
    <value id='diastolic' path='/entryRelationship[observation/code/@code="271650006"]/value/@value'/>
    <value id='units' path='/entryRelationship[observation/code/@code="271649006"]/value/@unit'/>
</contentbinding>
```


an example of a blood pressure measurement that was part of a CDA document example in the HL7 ballot of May 2008 [17] and the R-MIM model of a blood pressure as modeled in the Netherlands [18].

Figure 20 shows a binding to an archetype blood pressure. Figures 22 and 23 show it is possible to create a slightly different Binding Content Unit and reuse the Display Content Unit to display an HL7 structure.

Two issues emerged:

- In the CDA example the only distinction between the systolic and the diastolic part of the blood pressure could be made by including a test for a specified code in the path to the value. This implies that the code system is stable or the Binding Content Unit needs to be extended with a mechanism that allows a selection of the path based on the results of various tests for the code systems used. In the R-MIM model of the blood pressure, this issue is resolved by having a specific 'SystolicPressure' and 'DiastolicPressure' part. In Archetype Definition Language (ADL), this problem is solved by using stable internal references that are independent of the coding schemes used.
- In the R-MIM model, the values of the systolic and diastolic parts have no separate unit entry. From the definition of the type of the value, it was not clear if a unit was implied (e.g. through the coding scheme used), part of the value structure (but not stated separately) or even part of the content of the value item (i.e. '120 mmHg' rather than '120'). In the CDA sample a separate entry for the unit was available. To overcome such a situation, the Binding Content Unit needs to be extended with a mechanism to set the value to a fixed entry. Obviously, the better solution is to include a clear separate unit entry in the R-MIM model.

5 Discussion

5.1 APPLICABILITY OF THE APPROACH

There are several advantages to the solution proposed here:

- First and foremost, it offers the flexibility of defining specific, optimized screen presentations for known information structures, while providing the means to generate usable screen presentation of previously unknown information structures.
- By separating the various types of presentation knowledge into distinct models, it is possible to separate pure GUI knowledge from medical presentational knowledge, thus adopting another two-model approach and promoting reuse.
- It is flexible enough to build an adaptive GUI based on roles. For example, nurses and physicians can see the same information but optimally presented for their specific needs, while the only difference in development might be a document definition.
- The evolution of medical knowledge will always create new data types and new archetypes, which would lead to new display representations. This approach ensures that the available display knowledge can be reused as much as possible.
- New and novel ways to view EHR information are a topic of ongoing research. By adhering to the proposed approach, these views can benefit from the available display knowledge that is already expressed in content units [19]. In the blood pressure example earlier, the content unit for the single blood pressure can be reused to define a list of blood pressures that can be used in a view to display an interactive timeline of measurements as well as a simple list view. More specifically, by separating the Display Content Unit from the Binding Content Unit, the same Binding Content Unit of the blood pressure could be mapped onto a Display Content Unit specifying the common Systolic/Diastolic format or an X/Y format, where X is bound to the timestamp, Y1 to the systolic and Y2 to the diastolic values. By adding the appropriate calculation, the Y value could even be mapped onto the mean blood pressure. A Chart content unit can take the list of X/Y formats as input for producing the graph.
- Both the HL7 RIM and CEN 13606 archetypes use a limited set of predefined data types, with ongoing efforts to harmonize the two sets. By describing one or more widgets for each data type, it should be possible to provide a meaningful display of the information without incorporating presentation knowledge in the information structure. This means the current archetype or message specifications need not be extended and the number of widgets remains small.

- The information can be converted to match local and user preferences such as preferred coding scheme, language, units and more.
- A fallback mechanism is used to select a more generic representation in the absence of a specific one.
- Standardized content units could be shared between systems the same way archetypes can be shared. This increases the intelligence and usability of the system.
- The pipeline approach not only allows reuse of components, but also offers flexibility in functionality by a simple addition of pipelines.

This approach complements the *open*EHR architecture where templates are used to create a higher-level composition by constraining and ordering of archetypes. However, the *open*EHR templates are used to create an information structure, while the approach proposed in this paper is used to display the structured information. There are also disadvantages:

- Specialized views, containing special content units, can only exist for predefined information types. New information types or information types from different domains will fall back to a more basic representation. The latter, however, can benefit from shared content units.
- A repository equal to that for archetypes is necessary for the various presentation units.
- A mechanism for retrieving an appropriate screen representation for the current archetype is necessary, since the most appropriate selection is based on multiple parameters that cannot be stored in the archetype instance.

From the proof-of-concept implementation two issues surfaced:

- *Value-related display.* Uniquely marking special values (e.g. out-of-range values) enhances quick interpretation of the data. This involves the union of several types of knowledge. The GUI designer, along with users, decides on the display of the special value (e.g. large red font, flashing and/or audible signal), the content modeller decides whether the out-of-range value needs to be marked (or not), while either the archetype definition or a separate knowledge base holds the information on when the value is out of range. Complexity increases when a range has several sub ranges with different interpretations, for example the BMI range can be divided in underweight, normal, overweight and obese. The content definition language needs to provide a method of indicating when and how a special value needs to be marked as well as a reference to the location of the underlying knowledge. All this should be combined with a consistent, yet appropriate display (e.g. always red for alerts, unless the concept requires otherwise).
- *Conversion rules.* Users need to be able to specify their preferred units, local-

ized date formats etc., because even when the unit of measure is displayed, sometimes the value in itself looks 'unfamiliar' at first glance thus hindering immediate interpretation. This requires an extensive conversion library, which needs to be implemented in a generic way. It is not clear where this implementation should be located when the conversion extends beyond the universal conversions (e.g. Celsius to Fahrenheit) and enters the clinical domain (e.g. the BMI calculation). The latter would ideally be located in the archetype, but there is currently no suitable expression language defined.

More work needs to be done to define a flexible content definition language that solves these issues and is capable of describing domain-related display knowledge at a sufficient level.

The advantages of having flexible GUI interfaces outweigh the disadvantages. By incrementally defining screen presentations that can be built on top of each other, there is less duplication of work in building a GUI. A higher-level screen presentation allows the user to better interpret the presented information leading to more efficient and more reliable information exchange. Screen presentations of new information structures can be added to the system without major redevelopment of the application.

Moreover, less effort is needed to define content units compared to the effort of defining archetype definitions, since an existing archetype definition can be used as the basis for the content unit definition. Since many medical concepts can be displayed using generic content units such as a simple label, value, unit triplet, the majority of concepts can be defined using a limited set of generic content units.

5.2 REQUIREMENTS

From the implementations of the three frameworks (CForms, XUL and proprietary), the authors draw several conclusions that elaborate the requirements of valid content definition languages. The implementation also led to proposed enhancements for the archetype structures.

5.2.1 Requirements for a content model language

The complexity of an archetype model is usually hidden behind an application (for example, the archetype editor from Ocean Informatics [20]). It would also be possible to hide the complexity of the content model language but it should nevertheless be relatively easy to understand for domain experts. It must have several functionalities:

- Description of positional relations between values and fixed information, for example in the case of a blood pressure format ("S/D") where the entire struc-

ture is horizontally oriented and the / sign is fixed and relevant.

- Description of optional parts of the content widget, for example the label is optional if a list of similar values is displayed and the list already has an appropriate header. This makes it possible to reuse the widget on a small screen such as a cell phone screen.
- A mechanism that defines the display behavior when the value is within a specific range, for example when the BMI value is in the "obese" range it should be colored red. This requires a mapping to a location, which holds information on the meaning of the various ranges as well as a description of the display options for each of the ranges. Note that the description of the meaning of these ranges should not be part of the content unit but of the archetype definition or a specialized knowledge server.
- Description of a binding to the underlying archetype. This binding ties the label, value and unit to the respective counterparts in the archetype. The binding language should be generic enough to accommodate other structures than archetypes.

More work needs to be done to solve these issues and to define a flexible content definition language that is capable of describing domain-related display knowledge at a sufficient level.

5.2.2 Requirements for archetypes

The proof-of-concept implementation has shown that correct interpretation of an archetyped EHR instance (for example, the actual description of a blood pressure) cannot be done without the associated archetype, since some semantic information is not duplicated in the instance. This is also true for localization of the information. Archetypes support translations of a concept into various languages, but the instance only contains the requested language. If the receiver wants to use several languages then the archetype itself must be available to the proposed framework. This makes the functionality to retrieve the archetype from a repository that is available to all receiving systems, a requirement, not an option.

During the proof-of-concept implementation, we observed that the same information was defined using two different archetypes. A complete blood count report was coded with a specific CBC report archetype and with a more generic laboratory report archetype. In the former, each triplet of name, value and unit was uniquely identified, thus providing a path to each of the values, while in the latter, the triplets were merely part of a list without a unique identification at item level. The former instance allowed for a specific content unit controlling the order in which the test results were displayed, while the latter could only provide a more generic list in the order available in the instance. Although both structures were valid EHR instances,

the authors suggest that information should only be structured using the most specific archetype definition available. It is proposed to adjust the mechanism used in archetype selection during data entry so that the most appropriate archetype is selected more rigorously.

There is currently only a rudimentary version of an expression language defined in the *open*EHR Archetype Definition Language (ADL) [14] that can be used to define calculations based on archetypes or archetype parts. The current ADL version (1.4) describes how to reference archetype parts within the archetype (e.g. the systolic and diastolic values in a blood pressure archetype) but not across archetypes, e.g. the BMI calculation which combines the weight and height archetypes.

This is a restriction that is not necessarily applicable to other formalisms. Expressing an archetype in an imperative programming language would provide access to all the computational power of that language, including validations based on cross-archetype values.

Finally, in order to implement a fall back mechanism in the selection of the content units from specific to generic, it is desirable to have inheritance information in the archetype definitions. In the above example of the CBC report, examination of the specific CBC report archetype shows it is inherited from the more general laboratory report archetype. However, there is no required indication of this inheritance in either the EHR instance or the archetype definition. The ADL as well as the Archetype Object Model (AOM) [21] do provide an entry 'parent_archetype_id', but this is optional and the accompanying text does not clarify if this should be used to indicate the inheritance path, or to express which archetypes are allowed as parents.

This situation might be improved, if the entry were a required indication of the inheritance path, possibly adding a special value to indicate 'no-parent' or 'root-archetype'. Additionally the process of archetype definition should enforce the selection of the most appropriate parent.

The lack of this inheritance path does not invalidate the content unit approach. It merely results in a more generic display than necessary.

The requirements to the archetype model presented here are not unique for the proposed framework. Any GUI framework would benefit from them.

5.3 RELATED WORK

A similar approach has been developed by Ocean Informatics and implemented in their EhrView application [22]. The EhrView application modifies the information through a series of XSL stylesheets. These stylesheets are selected by matching the archetypes names in the structures. The matching process selects the most specific stylesheet available in a repository. The EhrView application does not separate con-

tent-related modelling from software-related modelling and it is only defined for one type of device: a regular screen of a desktop PC or laptop. It also offers limited options to adjust to local or user preferences.

Fiala et al. [23] have described a component-based approach for adaptive Web documents that influenced the approach reported here. They too make a distinction between content-related and display-related presentation knowledge and they used the pipeline concept to define Web document generation. However, their focus is on adapting information presentation to user preferences and devices. The information is known and defined in advance and there is no method to handle unknown information structures or describe conversions to preferred units.

University College London (UCL) uses a Java-based approach, and has developed their own rendering toolkit. Class files are annotated with Enterprise JavaBeans 3.0 (EJB 3) descriptors as well as directives to the toolkit, so that one single archetype file (including content often represented in ADL) can be used to create every tier of an application without further (archetype-related) effort being required.

Outside the healthcare domain, several studies on adaptive GUIs and UI modelling languages have been done (e.g. [24]). The majority of these studies either focus on the adaptability to different display devices (e.g. [25]) or to the disabilities of the user (e.g. [26]). They start from the premise that the information to be displayed is known, which is the problem embedded in the lower section of the GUI model in Figure 15. However, it is possible that some existing modelling languages could be made sufficiently expressive for a complete role in archetype visualisation.

5.4 FUTURE WORK

More research is needed to define a Content Unit Definition Language that is flexible, yet easy enough to use. This should be accompanied by a binding definition language that provides a simple mechanism to map a display definition to the associated archetyped EHR instance.

Although XUL could provide the initial version of the language to handle positions and orientations, it needs to be extended with a binding definition language that matches the ease of the CForms binding definition language.

The content definition language also needs to support descriptions of more complex content units. These require references to knowledge bases with reference information (such as the reference graphs in a pediatric growth chart) and flexible mechanisms to define highly specialized graphic structures for charts and family trees.

The Scalable Vector Graphics (SVG) specification is a modularized language for describing two-dimensional vector and mixed vector/raster graphics in XML.[27] It is a good candidate for the description of such graphic structures, but it is too

complicated to manually define them therein. It could be the basis for specialized widgets that can be used in the content units. This, however, should be done carefully to avoid convolution of the separation of the domains. In the example of the family tree, the tree is composed of persons and relationships. Relationships are described as lines and persons are depicted by shapes. Since colors, line styles and shapes all have meaning; it is difficult to separate the GUI-model knowledge (colors, line style, shape, fill color) from the Content model (meaning of relationship, gender, disease).

The authors have demonstrated that relatively little work is necessary to reuse this approach for HL7 structures. To fully evaluate the generic nature of the approach a more extensive set of archetypes and HL7 structures need to be tested.

5.4.1 DATA ENTRY

The approach presented here has been primarily focused on data consultation. While clearly data entry differs from data display insofar as one creates data and the data structures that contain it, and the other simply renders existing input, the layout should be shared between entry and display as much as possible. Doing this enables the technical rendering apparatus to be shared, and allows a clinician to later review data with a similar appearance to that of the original committer. However, this is non-trivial because data entry focuses on speed and data validation, not (so much) on appearance, as has been discussed by van der Meijden [28] and van Ginneken [29].

This approach can support data entry by extending the binding framework to binding events. Events can be used to perform calculations, data validation, entry support (e.g. context-related selection lists) and the creation of the data structure.

The generic workflow would be to retrieve existing data (in case of data modification) or an empty data structure (in case of new data) from the system kernel, which contacts the archetype server for the appropriate archetype references. This is passed to the GUI in the same pipeline process described earlier. Events that create data structures (e.g. a row in a table), can communicate with the system kernel on valid data structures. Finally, the resulting data structure is passed onto the system kernel for final validation and data storage.

Events to support data entry and data validation events need a closer connection to the validation parts of the archetype, to retrieve and provide valid subsets or verify if the entry is within the provided range. Connection to external knowledge sets (e.g. coding sets) is also necessary.

Event binding crosses the separation between the GUI model and the Content model separation because the events are widget-related, while the action is content-related. For example, pressing an "add new row" button is a simple GUI event.

The related action needs to communicate with the system kernel on how to extend the data structure to add a new row. This falls into the Content model domain.

6 Conclusion

Interoperability does not stop when information from one system can be successfully understood and/or incorporated in another system. It is also necessary to provide a screen representation that gives the user of the receiving system a clear understanding of the new information.

This paper describes a two-model GUI approach that extends that currently used by EN13606 for knowledge modeling. The authors argue that this approach leads to a flexible GUI that can adapt to information structures not predefined in a receiving system and display them in an interpretable way. This flexibility will also accommodate data entry since future-proof systems are in equal need of data entry forms for newly created concept structures. The work has demonstrated its generic nature by requiring only a minimal effort to map the framework to equivalent HL7 structures. It should be noted that a future-proof generic GUI methodology also places constraints on the archetype model.

Ongoing biomedical research in areas such as biomarkers and proteomics will eventually add new medical concepts to an EHR that need specific representations, not yet found in the current EHR systems. These will be able to take advantage of any user preferences for physical and other quantities already in being in the system automatically. Nevertheless, this work also cautions that promising frameworks might not hold out in actual implementation. The main reason for this is the underlying common assumption that (part of) the domain knowledge is hard-coded into the GUI.

7 References

1 Health Level 7 Inc. [last accessed 18 January 2008]; Available from: http://www.hl7.org
2 Health Informatics – Electronic health record communication - Part 1. Reference Model, CEN/TC251 prEN13606-1:2005:E, March 2005
3 Health Informatics – Electronic health record communication - Part 2. Archetypes, CEN/TC251 prEN13606-2:2005:E, December 2005
4 openEHR. [last accessed 18 January 2008]; Available from: http://www.openehr.org
5 Beale T. Archetypes – an interoperable knowledge methodology for future-proof information systems. Published on the internet 2000 [last accessed 18 January 2008]; Available from: http://www.openehr.org/shared-resources/publications/archetypes.html
6 Standards Coordinating Committee. IEEE standard glossary of software engineering terminology: IEEE Comput. Soc.(1990).

7 van der Linden H, Boers G, Tange H, Talmon J, Hasman A. PropeR: a multi disciplinary EPR system. Int. J. Med. Inform. 70 (2003) 149-60.
8 van der Linden H, Talmon J, Tange H, Grimson J, Hasman A. PropeR revisited. Int. J. Med. Inform. 74 (2005) 235-44.
9 Schuler T, Garde S, Heard S, Beale T. towards automatically generating graphical user interfaces from openEHR archetypes. Stud. Health Technol. Inform. 124 (2006) 221-6.
10 Cocoon Forms Block Implementation. [last accessed 2 June 2008]; Available from: http://cocoon.apache.org/2.2/blocks/forms/1.0/489_1_1.html
11 XML User Interface Language. [last accessed 18 January 2008]; Available from: http://www.mozilla.org/projects/xul/
12 van der Linden H, Grimson J, Tange H, Talmon J, Hasman A. Archetypes: the PropeR way. Medinfo 11 (2004) 1110-4.
13 Apache Cocoon Web Application Framework. [last accessed 18 January 2008]; Available from: http://cocoon.apache.org
14 Beale T, Heard S. Archetype Definition Language. (13 Mar 2007) [last accessed 4 June 2008]; Available from: http://www.openehr.org/svn/specification/TRUNK/publishing/architecture/am/adl.pdf
15 XUL: Template Guide. (22 February 2008) [last accessed 3 July 2008]; Available from: http://developer.mozilla.org/en/docs/XUL:Template_Guide
16 Bowers J. XUL Templates are a Waste of Time. (2004) [last accessed 3 July 2008]; Available from: http://www.jerf.org/resources/xblinjs/whyNotMozilla/notXulTemplates.html
17 HL7 Version 3 Standard Ballot Package Download. (2008) [last accessed 3 July 2008]; Available from: http://www.hl7.org/v3ballot/html/welcome/downloads/downloads.htm
18 Fleurke AM, Goossen WTF, Hoijtink EJ, van der Kooij J, Swen, Vlastuin M, et al. ALGEMEEN LICHAMELIJK ONDERZOEK: BLOEDDRUK. (2 September 2005) [last accessed 3 July 2008]; Version 1.0: Available from: http://www.zorginformatiemodel.nl/1_documentatie/Doc_Obs_Bloeddruk_R01_V1.0.pdf
19 Sundvall E, Nystrom M, Forss M, Chen R, Petersson H, Ahlfeldt H. Graphical overview and navigation of electronic health records in a prototyping environment using Google Earth and openEHR archetypes. Medinfo. 12 (2007), 1043-7.
20 Archetype Editor. [last accessed 2 June 2008]; Available from: https://wiki.oceaninformatics.com/confluence/display/TTL/Archetype+Editor
21 Beale T. Archetype Object Model. (20 Mar 2007) [last accessed 4 June 2008]; Available from: http://www.openehr.org/svn/specification/TRUNK/publishing/architecture/am/aom.pdf
22 EhrView. [last accessed 21 January 2008]; Available from: http://oceaninformatics.biz/Products/ProductDescription.pdf
23 Fiala Z, Hinz M, Meißner K, Wehner F. A Component-based Approach for Adaptive, Dynamic Web Documents. J. Web Eng. 2 (2003), 058-73.
24 Souchon N, Vanderdonckt J. A Review of XML-compliant User Interface Description Languages. DSV-IS 2003, (2003) p. 377-91.
25 Mitrovic N, Royo JA, Mena E, Luna MD. ADUS: Indirect Generation of User Interfaces on Wireless Devices. Seventh International Workshop Mobility in Databases and Distributed Systems (MDDS 2004), 2004.
26 Gajos K, Wobbrock J, Weld D. Automatically generating user interfaces adapted to users' motor and vision capabilities. Proceedings of the 20th annual ACM symposium on User interface software and technology (2007), 231-40.
27 Jackson D. Scalable Vector Graphics (SVG) 1.1 Specification. (14 Jan 2003) [last accessed 4 July 2008]; Available from: http://www.w3.org/TR/SVG11/
28 van der Meijden MJ, Tange HJ, Boiten J, Troost J, Hasman A. An experimental electronic patient record for stroke patients. Part 2: System description. Int. J. Med. Inform. 58-59 (2000), 127-40.
29 van Ginneken AM. Considerations for the representation of meta-data for the support of structured data entry. Methods Inf. Med. 42 (2003), 226-35.

(continued)

5

6

7

8

9

* Tuck inside

10

11

repeat step 6 to 10 on the other side

12

13

12x

14

Chapter 9

Discussion

Discussion

1 Summary of main findings

In chapter 1, the introduction to this thesis, three research questions were posed, which will be addressed in this section.

Is it possible to define a generic architectural framework for a domain agnostic EHR system?

This question can be answered positively, as demonstrated in the first part of this thesis, chapters 2 to 5. This framework consists of a componentized architecture in which generic functionality of the EHR system is grouped into clearly delineated components that provide the services necessary to support the EHR as defined by the ISO definition in § 1 of chapter 1.

Is it possible to implement such an architectural framework?

Chapters 3 to 5 have demonstrated that it is possible to implement such a framework into a working EHR system. The design and implementation, described in chapters 3 and 4 respectively, were evaluated in chapter 5. This led to the conclusion that the objective of building a generic EHR system has been achieved. This chapter demonstrated that, despite technical and organizational issues, the trial users were positive about the system. This chapter also demonstrated that the system was generic and domain agnostic enough to reconfigure it for various domains without touching the system code.

Does this framework remain valid over time?

The validity of this framework was studied by performing a literature review on the period after the PropeRWeb from 2004 – 2007. Chapter 6 discusses the trends in the Medical Informatics fields with regard to EHR system architecture that occurred after the implementation of the PropeRWeb system. From this chapter the conclusion can be drawn that the concept of a componentized architecture is still valid. Literature has shown that EHR system vendors have not undertaken the opportunity to redesign their systems using similar architectures, but that eHealth programs at a regional level or beyond naturally follow this approach since they start in a distributed environment.

2 Additional findings

The framework was also studied by widening the scope in other directions: across organizational borders and in domain independency and user interaction.
The issues that arise when exchanging information between organizations are discussed in chapter 7. This chapter focuses on the security aspects of the cross-organizational context. It states that a true virtual patient record requires more clarity on the implementation of authorization models, patient restrictions and patient consent as well as legal and computational frameworks for confidentiality protection. More important is the paradigm shift in the health care professionals' minds from a digital version of the old paper-based records where care is mostly based on locally available information (whether locally created or received through information exchange) to a true virtual patient record where location and ownership is irrelevant unless pertaining to the patient's health.

Chapter 8 builds on the context of information exchange between organizations. In such an environment, the chance of receiving information that is not predefined in the receiving system is significant. Even if the receiving system is able to manage the information, it is not by definition capable to display the information in a sensible manner for the user. This chapter explores this situation and proposes a framework for a generic GUI that still honors the two-model approach by separating the concept knowledge, as defined in archetypes, from the display knowledge, which is defined in a separate display model.

Both chapters provide additional requirements for a true future-proof, interoperable, generic EHR system.

3 Methodological considerations

One of the major strengths of this project was the implementation of the framework in a working EHR system. The other major strength of the project was the focus on the generic aspect of the system.
At the time, it was one of very few archetype-based EHR systems deployed in a working environment. To date it still is the only archetype-based system available that maintains strict domain independence.
This implementation revealed several issues:

- Technical issues concerning the completeness of the specifications and the use of open source.

- Organizational issues regarding the environment in which the system was deployed.
- User participation in the development process of the system.

Each of these issues will be addressed below.

3.1 TECHNICAL ISSUES

The specifications of the archetypes and related components at the time of the design of the PropeRWeb framework suggested that a complete set of components was available for a new generation of EHR systems. During design and implementation, it became evident that an archetype kernel was not sufficient. Several additional components were necessary:

- A component that handles the storage and retrieval of the object instances that are generated from the archetypes (archetype querying was mentioned often, but only very recently a query language was defined [1]).
- A component that handles filtering of unauthorized information based on security policies.
- In addition, no GUI related components were available for display and data entry of archetype-based information. The focus was on the archetype editor, not on an environment to use the archetypes to create object instances.
- Finally, there were no predefined archetypes available in general and certainly not for the CVA domain. This also meant that not much expertise on modeling archetypes was available, which made it impossible to guide future users in the process of modeling CVA related archetypes and resulted in a conversion of the paper based forms into simple archetype-like structures.

At the time of design and implementation of PropeRWeb, CORBA was an important standard for middleware connectivity. This, and the fact that the CorbaMed components were based on CORBA made the choice for CORBA as middleware obvious. In recent years, SOAP and Web Services have replaced CORBA as the de facto standard in connectivity, which would warrant any future implementation of PropeRWeb to be based on Web Services.

3.2 OPEN SOURCE

The use of open source components from various projects has been demonstrated to be advantageous in several ways, as described in chapter 3 and chapter 5. Some advantages and disadvantages of the use of open source will be summarized here:

- The drawbacks of limited development resources were compensated by the available functionality of the software of the open source projects as well as by the help from the various communities.

- Open source reference implementations of standards and/or specifications, such as done by the OpenEMed project [2], not only show the feasibility of the implementation but also demonstrate the validity of the design, and more importantly, reveal possible ambiguities and omissions, that are often not apparent from the design only.
- Open source provides an easy way to verify compliance of the software to standards and regulations; an important advantage in a sensitive domain such as health care.
- Most open source projects are started to solve a problem of the initial developers. This results in two observations:
    - The developers are the "owners" of the project and are highly motivated to solve their problem. In health care IT, the health care professionals have a problem that needs to be solved by developers, which means motivation to solve the problem has to be conveyed from the care professional to the developer. This may be the underlying reason that communities of open source health care projects are much smaller than those of projects with a more generic goal.
    - Software development is focused on solving the problem, which may not align with the complete implementation of the available specification or standard. An example of this situation was the OpenEMed PIDS component, which lacked the implementation of the composite traits (see chapter 5). This problem is countered by contributions of others that joined the community.
    - The use of open source might be beneficial in terms of costs and development effort, but it might still require a steep learning curve to get insight in the available functionality and to get an overview of how the software matches the requirements of the project it will be used in.

The use of software from open source projects has led to the conclusion that open source intrinsically promotes reuse of code and therefore a more efficient use of collective intellectual property. Development of vital EHR system components as an open source joint effort of all EHR system vendors would not only reduce development costs for the respective vendor but the combined financial efforts would exceed any budget a single vendor can raise. More importantly, the resulting components naturally provide a strong basis for integration of information.

The result of open source projects is not just software. The communities that develop the software demonstrate a different mindset towards the software that is not based on individual ownership but on individual contributions to a shared code resource.

By adopting this mindset in the care process of a patient, the information would be shared more easily with the healthcare professionals concerned with a focus on the health of the patient, rather than the ownership of the information.

3.3 ORGANIZATIONAL ISSUES

Initially, the PropeR project set out to provide electronic support of the clinical CVA project to improve the quality of care in the rehabilitation of CVA patients in their home environment.

The organizational changes resulting from the clinical CVA project were already in place and appeared to be functioning. This looked like a stable environment to add the changes necessary for the PropeR project. However, in the course of the development of the PropeR project, several changes were implemented by the clinical CVA project without informing all parties involved. This seriously hampered the PropeR project. It is impossible to measure the quality of change in an environment that is changed by external factors as well.

Although the clinical CVA project and the PropeR project worked towards the same goal, the PropeR project was not taken into account by the clinical CVA project, which also led to the aforementioned unnotified changes. Clear agreements and mutual support between the projects should have been implemented and would likely have led to a result that is better than the sum of its components.

There were technical problems, but more important for success or failure are the organizational issues. Without support and active participation, the chances of success are slim.

3.4 USER PARTICIPATION

The common method of involving users in the design of the software was followed. Very early in the project, users were asked to give their opinion on how the application should function. Mock-ups of the screens, done with a simple drawing application, were demonstrated and reviewed. Feedback was incorporated in new revisions. The actual design and implementation of the software however, was done outside the view of the users. There was only one developer, which led to a considerably long period of 'silence' for the users. This in turn diminished their motivation to participate.

Development should be done with a larger team. This not only diminishes the development time, but also increases the quality of the work as well through better team feedback.

Development of a generic framework should also be done ahead of user participation.

This framework should be based on the initial framework, described in chapters 3 and 4, and combined with the GUI framework described in chapter 8. The rudimentary implementation of archetypes, as described in this thesis, should be replaced by a standardized reference model and complemented by appropriate development tools such as archetype editors and GUI editors.
Since the software is focused on configuration rather than compilation, the software development could be in the final stages before the users are asked to comment.

4 Related research

Several projects have performed related research. A few will be discussed here.

4.1 OPENMRS

OpenMRS is "a multi-institution, nonprofit collaborative led by Regenstrief Institute, Inc., a world-renowned leader in medical informatics research, and Partners In Health, a Boston-based philanthropic organization with a focus on improving the lives of underprivileged people worldwide through health care service and advocacy." [3]
The OpenMRS software is an open source web based EHR system based on a database model developed by Regenstrief and a concept dictionary. The system is complete and used in various developing countries. It is similar to PropeRWeb in that it also allows easy configuration of the system for different domains and provides both data entry as well as data retrieval. Unlike PropeRWeb it is not a generic system, since the data model contains several domain related concepts such as 'drug', 'order' and 'patient'. The concept dictionary has the same function as the PropeRWeb archetypes: avoid hard-coded implementation of domain concepts in the database.
Study of the content of the concept dictionary shows that archetypes are more flexible, since they not only provide a concept with a description, but also provide more metadata such as unit of measurement and terminology codes. Part of the concept in the concept dictionary is links to related concepts to build a hierarchy. Archetypes lack this type of links. However, binding of concepts with a terminology system like SNOMED/CT would provide a similar functionality.

4.2 OPENEHR

*open*EHR [4] has built on the results of several EU and Australian projects, primarily the Good European Health Record (GEHR), Synapses, Synex and the Good Electronic Health Record (GeHR) project. *open*EHR has proceeded in two directions:

further development of the specifications of archetypes and related work and the development of open source components that provide the software implementation of the specifications. Ocean Informatics [5] is one of the founding partners, and is one of the main contributors to the development of the specifications while building commercial implementations of the specifications.
Over the years, the archetype specifications have been refined based on community and software development feedback. As an example: the initial archetype specifications did not support multiple languages, which was added later. The need to provide display and data entry for archetype-based information has led to the development of templates that group related archetypes as well as allow defaults for various attributes. These templates provide comparable functionality of the profiles described in the GUI framework of chapter 8.

4.3 ARTEMIS

The Artemis project [6, 7] focused on integration of multiple EHR systems based on the IHE Profiles [8] and Web Services. An important issue studied by this project was the mapping of the various terminologies and ontologies used in the various systems. This issue has not been considered in the PropeRWeb framework and needs further study.

5 Future research

Although this thesis has demonstrated that the PropeRWeb framework is generic enough to be easily configurable for different domains, there are still some issues unresolved. As demonstrated in chapter 8, genericity should be extended to the GUI. In the current implementation, no security other than a simple authentication mechanism was implemented. Security mechanisms affect information in two ways:

- Data that should be hidden from the user
- Functionality that should be hidden from the user

Both have an impact on the GUI. This leads to the following research question:

> *How can standards-based security be implemented in a generic EHR system in such a way that the domain independency is maintained?*

Looking at the framework from a higher level, the level of regional integration of various systems, the integration of various domains becomes more prominent. To properly manage such an environment the current framework needs to accommo-

date for multiple clinical data sources and multiple demographic data sources. This will lead to two challenges:

- Matching of different demographic data of patients
- Mapping different terminologies used in the various clinical data sources

Since the first challenge has already been identified in numerous system integrations at hospital level, ample literature is available and several solutions have been proven more or less successful. Even the PIDS specification includes a correlation manager to map different demographic sets to a single person. Section 4.2.1.1 of chapter 7 addresses this issue as well. Extension of the PropeRWeb framework with a correlation manager component would provide a big step towards the solution of this challenge.

Mapping of different terminologies is a bigger challenge. Projects like UMLS [9] and GALEN [10] have addressed this problem, but to date no solution has been provided. Applying this knowledge to the PropeRWeb framework, more research needs to be done to answer the following question:

> *How can standards-based terminology mapping be implemented in a generic EHR system while maintaining domain independency?*

Note that in this question the EHR system presents itself to the user as a single system, while it integrates information from various systems. This terminology mapping is implemented in two aspects of the system: a terminology server that is capable of providing the appropriate code of a different terminology system, which requires the enormous effort of mapping all codes of various codes systems to all other code systems, and the support of several terminology sections in archetypes. The latter seems a smaller effort since the archetype defines a single concept that is clearly delineated.

The importance of archetypes is increasing, which requires a more formal approach to the definition of archetypes. This approach should ensure good quality archetypes. The latter can primarily be described as archetypes that are widely recognized by medical professionals as a correct description of the underlying medical concept. Good quality archetypes also have clear boundaries in their relation to other archetypes as well as support semantic interoperability as much as possible. The need for such a formal approach is now recognized and has lead to work done on defining governance, validation and certification processes. [11]

6 Concluding remarks

In hindsight, it can be argued that the system was too advanced for the technology at the time. Although GPRS coverage of the region was said to be complete, the test users experienced many locations where no signal was available. Over the test period, general use of GPRS increased, therefore limiting the speed of the individual users. This was experienced by the test users by longer page loads and a higher frequency of timeouts.

Recently GPRS is replaced by UMTS that provides a higher speed. In addition, WiFi is almost ubiquitously available with lower costs and higher speed. Viewing web pages on mobile devices, even smaller than laptops, is becoming common practice, resulting in more technical infrastructure and optimized software to support it. We can only conclude that using innovative technology, even if it seems stable to use, should not be used for critical applications that need to be available 24/7.

The timeout problems were partially due to the sub-optimal performance of the first version of the software. This problem did not occur with the second version, but since the number of patients had dropped, the second version has not been used as much as the first version.

The system was also advanced in another aspect: the focus on building a complete EHR system based on archetypes, while participants in the *open*EHR community focused on systems to support clinical models based on archetypes [5].

Finally, the development team consisted of a single person. This resulted in long periods of progress invisible to the users and waning enthusiasm.

Software development should be undertaken by several active developers. This would not only decrease development time, but also result in a high quality of the software (i.e. an implementation error such as the problem with the form modes described in chapter 5 would have been revealed earlier).

This project has shown that the common approach of user involvement in the design phase followed by a development phase can only work if the development time is relatively short. Several researches have shown that several rounds of user involvement in the software are necessary to elicit the explicit user requirements. Since the software was intended to be generic, we believe that a reversal of user participation and development would have been more successful. Rather than developing the design consented to by the users, the users would have been demonstrated a prototype application built in the PropeRWeb software, which would have lead to changes that could have been implemented in a short time span.

This project has shown that security should be implemented in the design from the beginning, not added as a separate component afterwards, even if it is implemented in a separate component.

The objective of this study was the feasibility of a development framework for a generic EHR system, based on a component based architecture as well as a dual model approach, using available standards and open source as much as possible and while keeping the system as domain agnostic as possible. The conclusion is that such a framework can be developed and provides a good foundation for generic EHR systems, as demonstrated in this thesis. It is also demonstrated that not all aspects of such a framework have been resolved and further research in several directions, such as security, terminology and GUI-definition, all with the underlying domain agnostic aspect, is needed.

7 References

1 Ma C, Frankel H, Beale T, Heard S. EHR query language (EQL)– a query language for archetype-based health records. Stud Health Technol Inform. 2007;129(Pt 1):397-401.

2 OpenEMed. [cited December 2002]; Available from: http://www.openemed.org

3 OpenMRS Overview. [cited 2009/02/03]; Available from: http://openmrs.org/wiki/OpenMRS_ Overview

4 OpenEHR. [cited 4 November 2002]; Available from: http://www.openehr.org

5 Ocean Informatics. [cited 2009/01/21]; Available from: http://www.oceaninformatics.com/

6 Boniface M, Watkins ER, Saleh A, Dogac A, Eichelberg M. A secure semantic interoperability infrastructure for inter-enterprise sharing of electronic healthcare records. Stud Health Technol Inform. 2006;120:225-35.

7 Dogac A, Laleci GB, Aden T, Eichelberg M. Enhancing IHE XDS for federated clinical affinity domain support. IEEE Trans Inf Technol Biomed. 2007 Mar;11(2):213-21.

8 Integrating the Healthcare Enterprise. [cited 2009/01/26]; Available from: http://www.ihe.net/

9 Unified Medical Language System. [cited 2009/02/03]; Available from: http://www.nlm.nih.gov/research/umls

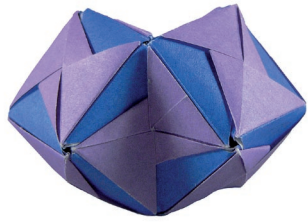
10 OpenGALEN Mission Statement. [cited 2009/02/03]; Available from: http://www.opengalen.org/

11 De Moor G, Kalra D, Devlies J. Certification of Electronic Health Record systems and the importance of the validation of clinical archetypes. Stud Health Technol Inform. 2008;141:82-91.

Appendices

Summary
Samenvatting

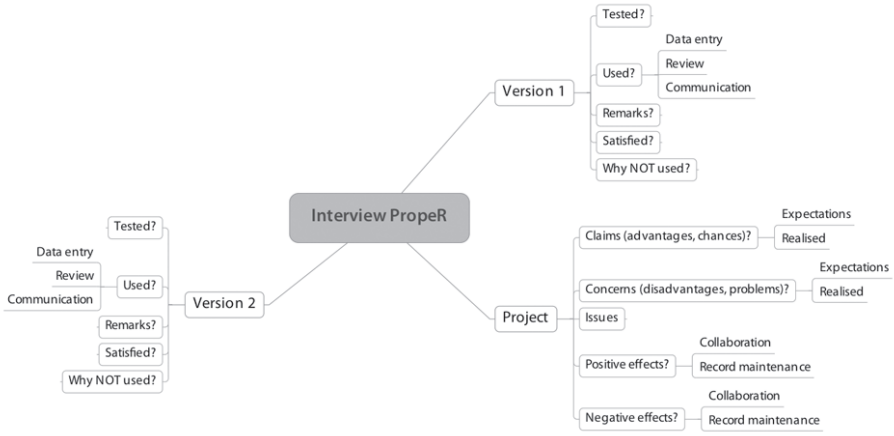
Curriculum Vitae
Publicaties
Dankwoord



Appendices

A Mindmap used in interview

FIGURE 24 Mind map of the interview



This appendix shows the mind map that was used to structure the interview of the four test users.

B GUI configuration files

This appendix shows an example of the various configuration files as used in version 2 of PropeRWeb, as described in chapter 5.

1 Archetype definitions

The archetype definitions are defined in the *archetypes.xml* file. The excerpt in figure 25 shows two archetypes defined: a date type with a predefined storage and retrieval format of the date indicating the transfer date of the patient and a list of predefined values to indicate if the patient has signed the consent form. Each archetype definition also has a *cocoon* attribute to reduce the complexity of the archetype name in the Cocoon environment.

FIGURE 25 Archetypes.xml

```
<elements>
…
    <element name="DNS:proper.mi.unimaas.nl/StrokeMeeting/TransferDate" cocoon="strokeMeetingTransferDate">
      <dateType format="yyyyMMddTHHmmss"/>
    </element>
    <element name=" DNS:proper.mi.unimaas.nl/StrokeMeeting/SignaturePatient" cocoon="strokeMeetingSignaturePatient">
      <rangeOrType>
          <item value="TRUE"/>
          <item value="FALSE"/>
          <item value="UNKNOWN"/>
      </rangeOrType>
    </element>
…
</elements>
```

2 Form definition

The form that defines the composition of all archetypes involved in the stroke meeting is defined in the *strokemeeting.xml* file. The excerpt in figure 26 shows the domain name of the form along with a simplified version of the name for the Cocoon environment. The label is used in the GUI, a.o. in the section labeled 3 in Figure 11. The *sortcode* element can be used to sort the forms in this section into an order that is more logical to the users. Finally, the archetypes are referenced.
The label should have been part of the presentation layer configuration files, which is also supported by the fact that it is the only item of this definition subject to localization issues. However, the current Cocoon CForms syntax does not allow this kind of labels at the form level.

FIGURE 26 Strokemeeting.xml

```
<form>
    <code>DNS:proper.mi.unimaas.nl/StrokeMeeting</code>
    <label>Overdrachtformulier</label>
    <sortcode>01-strokemeeting</sortcode>
    <cocoonname>strokemeeting</cocoonname>
…
    <element name="DNS:proper.mi.unimaas.nl/StrokeMeeting/TransferDate"/>
    <element name="DNS:proper.mi.unimaas.nl/StrokeMeeting/SignaturePatient"/>
</form>
```

3 Form display data entry widget definitions

The file *strokemeetingDef.xml* (see figure 27) contains the widget definitions for the form and the contained archetypes. The definition follows the Cocoon CForms syntax. Since these files are used in the Cocoon environment, the simplified names are used. The *transfer date* widget shows the display and data entry format of the date, the valid range and the possible failure message. Note that this date format differs from the storage format. The *signature patient* widget shows the mapping of the predefined values to a display description.

Various elements can be localized to allow for several language versions. For the sake of simplicity, this has not been done. Since Cocoon supports localization through the I18N block [28], the effort to add localization consists of changing the elements subject to localization, adding the appropriate key-message pairs for each language and the modification of the relevant Cocoon pipelines to add the I18N transformation.

4 Form display data entry widget binding

The file *strokemeetingBind.xml* (figure 28) contains the definition for the binding of the data entry widgets to their domain model counterparts. For this customized binding code was developed.

5 Form display data entry template

The *strokemeetingAddTemplate.jx* file (figure 29) holds the definition of the layout of the webpage. The *cinclude* items define the global sections of the page such as the user information, the list of available forms and the demographic information of the selected patient.

FIGURE 27 StrokemeetingDef.xml

```
<fd:form xmlns:fd="http://apache.org/cocoon/forms/1.0#definition">
  <fd:widgets>
…
    <fd:field id="strokeMeetingTransferDate" required="false">
      <fd:label>Ontslagdatum</fd:label>
      <fd:datatype base="date">
        <fd:convertor>
          <fd:patterns>
            <fd:pattern>dd-MM-yyyy</fd:pattern>
          </fd:patterns>
        </fd:convertor>
        <fd:validation>
          <fd:range min="Date(1900, 1, 1)" max="Date(2010, 12, 31)">
            <fd:failmessage>Voer een datum in tussen 1900 en 2010</fd:failmessage>
          </fd:range>
        </fd:validation>
      </fd:datatype>
    </fd:field>
    <fd:field id="strokeMeetingSignaturePatient" required="false">
      <fd:label>Toestemming patient</fd:label>
      <fd:datatype base="string"/>
      <fd:selection-list>
        <fd:item value="TRUE">
          <fd:label>Ja</fd:label>
        </fd:item>
        <fd:item value="FALSE">
          <fd:label>Nee</fd:label>
        </fd:item>
        <fd:item value="UNKNOWN">
          <fd:label>Onbekend</fd:label>
        </fd:item>
      </fd:selection-list>
    </fd:field>
…
  </fd:widgets>
</fd:form>
```


6 Form display data viewing template

The *strokemeetingViewTemplate.jx* file (see figure 30) defines the web page layout for the data retrieval page. Apart from the actual display of the elements, it is very similar to the template for data entry.

FIGURE 28 StrokemeetingBind.xml

```
<fb:context xmlns:fb="http://apache.org/cocoon/forms/1.0#binding" path="/">
…
  <fb:custom id="strokeMeetingTransferDate" path="elements/strokeMeetingTransferDate/value"
      class="nl.unimaas.mi.proper.coas.CustomCoasBinding"/>
  <fb:custom id="strokeMeetingSignaturePatient" path="elements/strokeMeetingSignaturePatient/value"
      class="nl.unimaas.mi.proper.coas.CustomCoasBinding"/>
</fb:context>
```

FIGURE 29 StrokemeetingAddTemplate.jx

```
<jx:template xmlns:jx="http://apache.org/cocoon/templates/jx/1.0"
  xmlns:cinclude="http://apache.org/cocoon/include/1.0"
  xmlns:i18n="http://apache.org/cocoon/i18n/2.1"
  xmlns:ft="http://apache.org/cocoon/forms/1.0#template"
  xmlns:fi="http://apache.org/cocoon/forms/1.0#instance" >
  <jx:import uri="context:/properweb/content/coastemplates/simpleListAddMacros.jx"/>
  <layout>
    <head/>
      <cinclude:include src="cocoon:/getUser"/>
      <cinclude:include element="sidebar" src="cocoon:/formlist"/>
      <cinclude:include src="cocoon:/getSelectedPatientInNavbar"/>
      <jx:choose>
        <jx:when test="${action == 'add'}">
          <jx:set var="action_label" value="add"/>
        </jx:when>
        <jx:otherwise>
          <jx:set var="action_label" value="edit"/>
        </jx:otherwise>
      </jx:choose>
      <title>Proper <i18n:text>${label}</i18n:text> ${coasform.label}</title>
      <header/>
      <content addTopLink="true">
        <contentHeader><i18n:text>${label}</i18n:text> <i18n:text>form</i18n:text>: ${coasform.label}</contentHeader>
        <contentText><i><i18n:text>required_fields</i18n:text></i></contentText>
        <message>${errMsg}</message>
        <ft:form-template action="#{$continuation/id}.continue" method="POST">
…
          <fi:group>
            <fi:styling type="fieldset" />
            <fi:label>Medisch</fi:label>
            <fi:items>
              <verticalList>
                <col>
…
                  <headerRow elem="strokeMeetingTransferDate"/>
                  <headerRow elem="strokeMeetingSignaturePatient"/>
                </col>
              <data>
                <row>
…
                  <elementRow elem="strokeMeetingTransferDate" type="date"/>
                  <elementRow elem="strokeMeetingSignaturePatient" type="dropdown"/>
                </row>
              </data>
            </verticalList>
          </fi:items>
        </fi:group>
…
      </ft:form-template>
    </content>
  </layout>
</jx:template>
```

FIGURE 30 StrokemeetingViewTemplate.jx

```
<jx:template xmlns:jx="http://apache.org/cocoon/templates/jx/1.0"
   xmlns:cinclude="http://apache.org/cocoon/include/1.0">
   <jx:import uri="context:/properweb/content/coastemplates/simpleListViewMacros.jx"/>
   <layout>
      <head/>
      <cinclude:include src="cocoon:/getUser"/>
      <cinclude:include element="sidebar" src="cocoon:/formlist"/>
      <cinclude:include src="cocoon:/getSelectedPatientInNavbar"/>
      <title>Proper ${form.label}</title>
      <header/>
      <content addTopLink="true">
         <contentHeader>${form.label}</contentHeader>
         <message>${errMsg}</message>
         <simpleList>
…
            <row>
               <groupheader>Algemeen</groupheader>
            </row>

            <jx:set var="elemDef" value="${form.elements.strokeMeetingHomeSituation}"/>
            <elementRow elem="${elemDef}" header="Woonsituatie"/>

            <jx:set var="elemDef" value="${form.elements.strokeMeetingTransferDate}"/>
            <elementRow elem="${elemDef}" header="Ontslagdatum" type="date"/>

            <jx:set var="elemDef" value="${form.elements.strokeMeetingSignaturePatient}"/>
            <elementRow elem="${elemDef}" header="Toestemming patient"/>
         </simpleList>
      </content>
</layout>
</jx:template>
```

C Scenario

A patient, Mr. Jones, lives in a town with a large hospital, a small psychiatric institution and several general practitioners practices. These are all separate organizations with contracts to share relevant information. The patient has one specific GP whom he regularly visits and who is fully informed about the patient's medical history.
Mr. Jones has a history of depressions that once resulted in a short stay at the psychiatric institution (STEP 1). He is doing well now and his current medication prevents relapse into depression. Mr. Jones has given his permission to the psychiatric institution to respond to requests for information only from his GP (STEP 2). The psychiatric institution has sent discharge information to the GP (STEP 3). Mr. Jones has also informed his GP that he doesn't want his psychiatric records be disclosed to others, unless it might have serious implications on future treatment (STEP 4).
One day Mr. Jones develops a rash and consults his GP (STEP 5). The GP is unsure whether it is an allergic reaction or something else. Rather than investigating the issue himself, he decides to refer Mr. Jones to the dermatology department at the hospital (STEP 6).
The dermatologist wants to know whether there is medical information about the patient elsewhere, regarding allergies and medication. The answers of the patient are vague, so he decides to ask the GP[17],[18] (STEP 7), who responds in compliance with the patients wishes (STEP 8). The dermatologist orders also a blood test and a skin allergy test to be performed at the hospital labs (STEP 9). The lab performs the tests and sends the results to the dermatologist (STEP 10). The skin test reveals a mild allergic reaction to cats and the blood test results show slightly elevated levels.
One day later the lab sends new blood test results (STEP 11). The machine turned out to be at fault and was re-calibrated and the tests were redone. The new results are all within normal ranges.
In the next consultation, the dermatologist advises the patient to stay away from cats, and prescribes tablets to reduce the itch (STEP 12). Mr. Jones is referred back to the GP (STEP 13).

17 We assume the patient has given consent to the dermatologist to request information from the GP.

18 It might be reasonable to assume that it is good practice to include this information in the referral letter. For this scenario, we assume it has not been done.

D Questions

The entire list of questions, grouped by scenario step.

Step 2*: The patient informs the psychiatric institution to restrict sharing of PSYCH information to only the GP.*

1 How should a patient be identified reliably across organizations?
2 How should health professionals be identified reliably across organizations? How should organizations be reliably identified?
3 Should all information (always) be available unless restricted by the patient or legislation or should information only be available if legislation and patient consent permit? At what level of granularity should the patient give his consent? Are there distinctions in types of situations? Should delegation be allowed, e.g. only if the GP thinks it is relevant?
4 Following from the previous question: Is it legally acceptable to ask for patient consent for access by unknown health-related external parties? I.e. *any* doctor or *any* nurse versus a specific, named person.
5 Can the patient consent to sharing his PSYCH information while consulting his GP? How should this be implemented?

Step 3*: Information from the PSYCH-EMR is shared with the GP (in compliance with the patient's consent).*

6 How should the PSYCH-EMR system define authorization of the GP to access information in the system?
6a How can the GP trust the information?
7 Should all systems have authorization information for all possible users (i.e. persons requesting information)?
8 Should all systems provide similar access for all possible users (i.e. a GP has access to the same kind of information in all systems)?
9 Dutch legislation (WGBO) only allows access to "need to know" information. Should systems be capable of deducing what information a GP needs to know, and if so, how can this be achieved?
10 If PSYCH-EMR information is stored in the GP-EMR system, how can the PSYCH-EMR system be informed of possible confidentiality breaches?

Step 4/8: The patient informs the GP to not share his PSYCH information.

11 In case the information from the PSYCH-EMR is stored in the GP-EMR system and matches a future query from an external system (e.g. the query from the DERM-EMR), should the information be passed on if patient consent permits or should external information always be excluded from a result set?
12 What happens with this type of restriction if the patient moves from one GP to another?
13 When an emergency override is necessary, what access restrictions have still to be obeyed?
14 Can the right of the patient to delete information from his EHR be sufficiently covered by a total access restriction?

Step 7: The dermatologist requests information regarding allergies and medication of the patient.

15 How should the query be propagated to all available systems in the area?
16 Should the dermatologist be able to request all available information or only **what** is relevant?
17 Should the dermatologist be able to access all available information of the patient (present in various systems) at any point in time or only **when** relevant?

Step 9: The dermatologist orders several tests.

Step 10: The lab returns the test results.

18 Is there a difference between the two implementations in how the dermatologist is informed?
19 If the information is assumed to remain with the owner, who is the owner? The person ordering the information or the person generating the information? Or both?

Step 11: The lab sends corrected test results.

20 How should the dermatologist be informed of the corrected information?
21 Should the system be able to log the fact that the dermatologist has actually seen the (updated) information?

Step 12*: The dermatologist sees the test results.*

22 Should the dermatologist be able to view both the current (i.e. new) and the old (i.e. those sent before the recalibration) values?

Step 13*: The dermatologist writes a discharge letter, which includes the results of the skin allergy test.*

23 Should the GP be able to differentiate between the various owners of the information included in the message?
24 Should the dermatologist be allowed to access the patient information after the discharge letter is created?
25 What action should be taken if the test results were updated after the discharge letter was sent?
26 What should be done with the data after legal data retention time has passed?

Summary

Electronic Health Record (EHR) systems store information on the health and health problems of patients. These systems have evolved from mere administrative systems to information sources supporting the care process. Most of these systems were built for a particular organization and/or a particular care domain. The ongoing trend towards multidisciplinary, integrated care requires a virtual EHR that combines health information of a patient from various information sources that cross the borders of organizations and care domains. The combination of this trend with the fast pace in which medical knowledge develops, calls for EHR systems that are capable of exchanging information with other health information systems, can be integrated with other systems to create a virtual all encompassing EHR and are flexible enough to accommodate new medical concepts. This thesis proposes and studies a generic EHR system framework that supports these requirements. An EHR system was developed and implemented to serve as test bed for the framework. This implementation has been evaluated.

Chapter 1 introduces the problem space of the PropeR project and defines four aspects of a generic EHR system that should be studied: functional requirements, architecture, standards and user interface. Requirements for such an EHR system framework should address these four aspects.

A literature review was done using these four aspects as guidelines. The review was based on literature in the timeframe 1994 – 2001, the timeframe preceding the start of the PropeR project. In chapter 2, the results of this literature review are presented. It concludes with a list of requirements for a generic EHR system framework. Such a system should be based on architectural as well as privacy and security related standards. These standards not only provide a more generic interface for information exchange and integration, but also lead to a more generic domain-independent framework. Most relevant for the framework are the CEN ENV 13606 draft standard, the HL7 v3 standards and the OMG CORBA and CorbaMed (later Healthcare Domain Taskforce or HDTF) specifications.

In chapter 3, a design based on these requirements is presented. It describes a componentized architecture that is based on the two-model or dual model approach, which divides the design of the EHR system in a Reference Model, and a domain model. The Reference Model is generic and reasonably stable and suited for implementation in a database. It provides the building blocks that can be used to structure medical concepts in the domain model. The medical domain is more

volatile and subject to changes due to scientific discoveries. The medical concepts are modeled as constraints on these building blocks. Hence, such medical concept structures can be stored in the EHR system, even if the structures are developed after the development of the system. These concept structures are generally referred to as archetypes.
The components in the PropeR architecture are based on the OMG HDTF specifications. Separate components handle demographic and clinical information, which not only enhance privacy but also support easier integration of multiple information sources.

Chapter 4 presents an implementation of this architecture in the PropeRWeb EHR system. This chapter discusses the various components and their reference implementation in the OpenEMed project. The system was built in Java, using CORBA and was web based to allow for ubiquitous access. It was intended for use by a multidisciplinary team of primary care professionals such as physical therapists and speech therapists that help patients with a cerebrovascular accident (CVA) to rehabilitate in their home environment. The PropeRWeb EHR system was primarily used for communication between the team members.

Three evaluations of the PropeRWeb EHR system were performed: a use pattern analysis, a qualitative analysis of the usability of the system and a technical assessment. The results are presented in chapter 5. From the use pattern analysis, it can be concluded that the system was not used very much. The reasons were multiple: several organizational changes reduced the number of suitable patients. Simultaneously, the workload of the team members was increased by introducing multiple information systems that had to be kept current. Technical problems with the mobile Internet connection resulted in slow performance of the system. The usability analysis was performed by interviewing the test users of the system. Analysis of these interviews showed that the users were generally satisfied with the functionality of the system but were demotivated by the organizational and technical issues mentioned before. In an attempt to overcome the technical issues, a second version of the system was developed. This version improved the performance and usability of the system itself, while increasing the domain independency. The organizational issues remained unsolved.
The second version of the system was used to test the domain independency by creating additional applications: an MRSA-registration application, a clinical trial application and an EHR for Acute Myeloid Leukemia (AML) patients. The successful creation of these applications by persons not involved in the original development of the system demonstrated that the system was generic with respect to

domain independency. These developments also showed that new applications can be developed quickly even while special purpose tools for defining archetypes were lacking and the required archetype specifications had to be built in raw data format (XML files).

To further validate the approach, a second literature study was performed with the same four aspects and the same methodology of the literature study presented in chapter 2. This second study covered the timeframe 2004 – 2007, which was after the implementation and test of the PropeRWeb EHR system. The results of this literature study show that standards that were emerging in the timeframe of chapter 2 have now evolved to full standards. Also, the EHR data model (i.e. the structure of the information) and the EHR system (i.e. the software system that supports and manipulates the information) are more clearly separated. This is partly caused by the recognition of the validity of the two-model approach, which is demonstrated by the larger number of projects studying or implementing archetypes in EHR systems. It can be concluded that the design of the PropeR system framework as described in chapter 3 is valid. The implementation as described in chapter 4 could be made more up-to-date by adhering to current de facto standards such as web services to replace the CORBA middleware.

From our evaluation and validation of our PropeRWeb system, it became evident that security issues are still a major concern of users, in particular when health data is shared across organizational borders. Chapter 7 studies the security aspects of an integrated virtual EHR. The study is performed by stepping through a scenario that involves multiple care domains in the treatment of a patient. At each step, questions are formulated concerning the exchange of information and related security issues. Verification of the issues in literature result in a list of recommendations to clarify security and privacy issues using legal and technical frameworks.

Our framework supports semantic interoperability among EHR systems. Semantic interoperability entails that systems are able to meaningfully manage information that is obtained from other systems. The archetype approach makes it possible to store received information in a semantically meaningful way even though certain concepts may not have been handled before by the receiving system. However, semantic interoperability is not only on storage of data in a meaningful way, but also about presenting this information to the user in a semantically correct way. This is the subject of the study in chapter 8. This study results in a framework for a generic GUI that is based on the same two-model approach used in archetypes but extends the concept to the presentation layer. The framework distinguishes

between different kinds of display knowledge to design semantically correct presentation definitions for the archetypes presented to the system. Extension of the PropeR EHR system framework with this generic GUI framework would enhance its future-proof nature.

Chapter 9 presents the main conclusions of this study and contains the general discussion. In summary, the PropeR system framework defines a componentized architecture based on the two-model approach of archetypes. Studies have demonstrated that the framework itself is valid. Our PropeRWeb system was the first archetype-based implementation of a domain agnostic EHR system. Such a generic system was proven to be feasible. However, several technical and organizational issues prevented routine clinical use. A reimplementation based on current standards and technologies would alleviate the technical problems encountered.

The objective of the study presented in this thesis was the feasibility of a development framework for a generic EHR system, based on a component based architecture as well as a two-model approach, using available standards and open source as much as possible and while keeping the system as domain agnostic as possible. The conclusion is that such a framework can be developed and provides a good foundation for generic EHR systems, as demonstrated in this thesis. It is also demonstrated that not all aspects of such a framework have been resolved and further research in several directions, such as security, terminology and GUI-definition, all with the underlying domain agnostic aspect, is needed.

Samenvatting

Elektronische Patiënt Dossier (EPD) systemen bevatten informatie over de gezondheid en de gezondheidsproblemen van patiënten. De algemene Engelse term hiervoor is Electronic Health Record (EHR) systems. Deze systemen zijn geëvolueerd van administratieve systemen tot kennisbronnen die het zorgproces ondersteunen. Meestal werden deze systemen speciaal gebouwd voor een bepaalde organisatie en/of een bepaald zorgdomein. De huidige trend naar multidisciplinaire, geintegreerde zorg vraagt echter een virtueel EPD dat de gezondheidsinformatie van een patiënt uit verschillende bronnen integreert. Deze bronnen bevinden zich in verschillende organisaties en verschillende zorgdomeinen. Deze trend en de snelle ontwikkelingen in de medische wetenschap vragen EPD systemen die in staat zijn informatie uit te wisselen met andere systemen om een allesomvattend virtueel EPD te creëren terwijl ze flexibel genoeg zijn om ook nieuwe medische concepten te verwerken. In dit proefschrift wordt een generiek EPD systeem raamwerk voorgesteld en onderzocht dat aan deze eisen voldoet. Het raamwerk is getest door het ontwikkelen en implementeren van een EPD systeem. Deze implementatie is vervolgens geëvalueerd.

Hoofdstuk 1 stelt het probleem en de achtergrond van het PropeR project aan de orde en definieert vier aspecten van een generiek EPD systeem die bestudeerd moeten worden: functionele eisen, architectuur, standaarden en gebruikersinterface. Eisen die aan een dergelijk raamwerk voor een EPD systeem gesteld worden dienen aan deze vier aspecten te voldoen.

Op basis van deze vier aspecten is een literatuuronderzoek uitgevoerd over de literatuur in de tijdspanne 1994 – 2001. Deze tijdspanne beslaat de periode voor het begin van het PropeR project. Hoofdstuk 2 presenteert de resultaten van dit literatuuronderzoek. Het eindigt met een lijst van eisen voor een generiek EPD systeem raamwerk. Een dergelijk EPD moet gebaseerd zijn op zowel architectuur- als privacy- en beveilingsgerelateerde standaarden. Deze standaarden bieden niet alleen een generieke koppeling voor informatie-uitwisseling en -integratie, maar leiden ook tot een meer generiek, domeinonafhankelijk raamwerk. Voor dit raamwerk zijn de CEN ENV13606 standaard, de HL7 v3 standaarden en de OMG CORBA en CorbaMed (later hernoemd tot Healthcare Domain Taskforce of HDTF) specificaties het belangrijkste.

In hoofdstuk 3 wordt een ontwerp gepresenteerd dat op deze eisen gebaseerd is. Het beschrijft een architectuur die bestaat uit componenten en gebaseerd is op

de tweemodellen aanpak die het ontwerp van een EPD systeem verdeelt in een informatiemodel (Reference Model) en een domeinmodel. Het informatiemodel is generiek, vrij stabiel en geschikt voor implementatie in een database. Hieruit ontstaan de bouwblokken die als basis dienen voor het structureren van de medische concepten in het domeinmodel. Het medische domein is weinig stabiel en aan veel veranderingen onderhevig door de wetenschappelijke ontwikkelingen. Door het modelleren van medische concepten met behulp van deze bouwblokken ontstaan structuren die in het EPD opgeslagen kunnen worden, ook als ze gemaakt zijn nadat het systeem ontwikkeld en ingezet is. Deze concept structuren worden archetypes genoemd.

De componenten in de PropeR architectuur zijn gebaseerd op de OMG HDTF specificaties. Demografische en klinische informatie wordt door aparte componenten afgehandeld, wat niet alleen de privacy verbetert, maar ook de integratie van informatie uit verschillende bronnen vereenvoudigt.

Hoofdstuk 4 bespreekt een implementatie van deze architectuur in het PropeRWeb EPD systeem. De verschillende componenten worden besproken en hun referentie-implementatie in het OpenEMed project. Het systeem is gebouwd in Java en maakt gebruik van CORBA. Het is webgebaseerd om toegang vanaf verschillende locaties mogelijk te maken. Het systeem was bedoeld om een multidisciplinair team van eerstelijnsprofessionals zoals fysiotherapeuten en logopedisten te ondersteunen in de thuis revalidatie van patiënten met een cerebrovasculair accident (CVA). Het PropeRWeb EPD systeem was primair bedoeld als communicatiemiddel tussen de teamleden.

Het PropeRWeb EPD systeem is op drie manieren geëvalueerd: een analyse van het gebruik, een kwalitatieve analyse van de gebruiksvriendelijkheid en een technische analyse. De resultaten zijn beschreven in hoofdstuk 5. Uit de gebruiksanalyse bleek dat het systeem niet vaak gebruikt werd, om verschillende redenen. Verschillende organisatorische veranderingen verminderden het aantal patiënten, terwijl tegelijk de werkdruk verhoogd werd door het invoeren van verschillende informatiesystemen die allemaal bijgewerkt moesten worden. Daarnaast zorgden problemen in het gebruik van mobiel internet voor een trage werking van het systeem.
De analyse van de gebruiksvriendelijkheid werd gedaan door de testgebruikers te interviewen. Analyse van de interviews toonde aan dat de gebruikers in het algemeen tevreden waren over de functionaliteit, maar gedemotiveerd werden door de eerder genoemde organisatorische en technische problemen. Om de technische problemen aan te pakken is een tweede versie ontwikkeld. Deze versie verbeterde de werking en gebruikersvriendelijkheid van het systeem en was minder domein

afhankelijk. De organisatorische problemen konden niet worden opgelost.
De tweede versie van het systeem is gebruikt om de domeinonafhankelijkheid te testen door meerdere applicaties te bouwen: een MRSA-registratiesysteem, een applicatie voor een clinical trial en een EPD voor patiënten met acute myeloïde leukemie (AML). Deze applicaties werden met succes ontwikkeld door personen die niet bij de eerdere ontwikkeling betrokken waren. Hiermee werd het generieke aspect van domeinonafhankelijkheid van het systeem aangetoond. Verder werd aangetoond aan dat het mogelijk was om snel nieuwe applicaties te ontwikkelen, zelfs zonder speciale hulpmiddelen voor het definiëren van archetypes; de benodigde archetype specificaties moeten met de hand in XML bestanden beschreven worden.

Om de aanpak verder te valideren, werd een tweede literatuuronderzoek uitgevoerd op basis van dezelfde aspecten en dezelfde methodologie van het eerste literatuuronderzoek uit hoofdstuk 2. Dit onderzoek besloeg de periode 2004 – 2007, de periode na de implementatie en test van het PropeRWeb EPD systeem. De resultaten van dit onderzoek worden beschreven in hoofdstuk 6. Deze resultaten tonen aan dat de standaarden die in de onderzochte periode van hoofdstuk 2 in ontwikkeling waren, nu geëvolueerd waren tot geaccepteerde standaarden. Ook wordt er een duidelijker onderscheid gemaakt tussen het EPD datamodel (de structuur van de informatie) en het EPD systeem (de software die de informatie verwerkt). Dit wordt deels veroorzaakt door de acceptatie van de tweemodellen aanpak dat zich uit in een groter aantal projecten dat archetypes in EPD systemen bestudeert of implementeert. Op basis van deze resultaten kan geconcludeerd worden dat het PropeR systeem raamwerk, zoals beschreven in hoofdstuk 3, valide is. De implementatie zoals in hoofdstuk 4 beschreven wordt, kan gemoderniseerd worden door de CORBA middleware te vervangen door huidige de facto standaarden zoals web services.

De evaluatie en validatie van het PropeRWeb systeem toonde aan dat de beveilingsaspecten uiterst belangrijk zijn, zeker als medische gegevens tussen organisaties worden uitgewisseld. Hoofdstuk 7 bestudeert de beveilingsaspecten van een geïntegreerd virtueel EPD. Dit onderzoek is uitgevoerd door een scenario te beschrijven waarbij meerdere zorgdomeinen betrokken zijn bij de behandeling van een patiënt. Er werd stapsgewijs door het scenario gelopen en voor elke stap vragen geformuleerd over de informatie-uitwisseling en de bijbehorende beveiligingsaspecten. Vervolgens werden deze aspecten geverifieerd in de literatuur, waaruit een lijst van aanbevelingen volgde om de beveiliging en privacy aspecten te verhelderen met behulp van juridische en technische raamwerken.

Het raamwerk ondersteunt semantische interoperabiliteit tussen EPD systemen. Dit maakt het mogelijk dat systemen op een betekenisvolle manier met informatie uit andere systemen kunnen omgaan. De aanpak met archetypes maakt het mogelijk om binnenkomende informatie op een semantisch zinvolle manier te verwerken, zelfs als de concepten onbekend zijn voor het ontvangende systeem. Semantische interoperabiliteit slaat echter niet alleen op het opslaan van data op een zinvolle manier, maar ook op het presenteren van deze informatie aan de gebruiker op een semantisch correcte manier. Dit probleem is het onderwerp van hoofdstuk 8. Dit hoofdstuk beschrijft een raamwerk voor een generieke grafische user interface (GUI) die op dezelfde tweemodellen aanpak is gebaseerd als archetypes, maar het concept uitbreidt tot de presentatielaag. Het raamwerk maakt onderscheid tussen verschillende soorten presentatiekennis om een semantisch correcte presentatie definitie te ontwerpen voor de archetypes die aan het systeem worden aangeboden. Uitbreiding van het PropeR EPD systeem raamwerk met dit generieke GUI raamwerk maakt het nog toekomstvaster.

Hoofdstuk 9 beschrijft de belangrijkste conclusies van dit onderzoek en de algemene discussie. Samengevat: het PropeR EPD systeem raamwerk definieert een architectuur van componenten gebaseerd op de tweemodellen aanpak van archetypes. Onderzoeken hebben aangetoond dat het raamwerk zelf valide is. Het PropeRWeb systeem was de eerste implementatie van een domein agnostisch EPD systeem gebaseerd op archetypes. Het is aangetoond dat zo'n generiek systeem mogelijk is. Verschillende technische en organisatorische problemen verhinderden het gebruik in de dagelijkse praktijk. Herontwikkeling op basis van de huidige standaarden en technologieën zouden de meeste technische problemen oplossen.

In dit proefschrift wordt de haalbaarheid onderzocht van een ontwikkelmethode voor een generiek EPD systeem, gebaseerd op een componenten architectuur en een twee-modellen aanpak, waarbij zoveel mogelijk gebruik gemaakt wordt van bestaande standaarden en open source projecten, terwijl het systeem zo domein agnostisch mogelijk gehouden wordt. De conclusie is dat het mogelijk is om een dergelijke methode te ontwikkelen en dat het daar uit volgende raamwerk een goede basis is voor een generiek EPD systeem, zoals aangetoond in dit proefschrift. Ook blijkt dat niet alle aspecten van dit raamwerk zijn uitgewerkt en dat vervolgonderzoek nodig is op een aantal gebieden, zoals beveiliging, terminologie en GUI-definitie, waarbij het domein agnostische aspect niet uit het oog verloren wordt.

Curriculum Vitae

Helma van der Linden werd op 23 juni 1963 in Maastricht geboren. Ze behaalde in 1981 haar VWO-diploma aan het Stedelijk Lyceum te Maastricht. Helma behoorde tot de eerste lichting studenten die in 1981 aan de pas opgerichte Tolk/Vertalersopleiding in Maastricht startte. Ze rondde deze opleiding af met Engels als hoofdtaal, Frans als tweede taal en het Medisch/Technische gebied als specialisatie. Ze startte in 1985 een opleiding aan de HTS te Heerlen, waar ze in 1990 afstudeerde in de richting Technische Informatica met als afstudeeropdracht een simulatiesysteem van de boordcomputer van de Airbus A310 voor educatieve doeleinden.
Aansluitend werkte ze als systeembeheerder op de Vertalersopleiding, waarna ze vrij snel overstapte naar het Instituut voor Revalidatievraagstukken te Hoensbroek, als systeem- en netwerkbeheerder en wetenschappelijk programmeur. In 1995 startte ze bij de vakgroep Medische Informatica aan de Universiteit Maastricht, aanvankelijk als systeem- en netwerkbeheerder, maar later als wetenschappelijk programmeur. In die laatste hoedanigheid heeft ze meegewerkt aan de ontwikkeling van een aantal applicaties voor automatisch toetsen van studenten.
In 2001 nam ze naast haar rol als software engineer ook de onderzoekstaak over van het Transmurale deel van het PropeR project.
Inmiddels is ze als software ontwikkelaar in dienst getreden bij OmniHis.
Daarbuiten is Helma actief als secretaris en webmaster van de landelijke Origami Sociëteit Nederland.

Publications

- van der Linden et al. Generic screen representation for future-proof systems -- is it possible? There is more to a GUI than meets the eye. Comput Methods Programs Biomed (2009) Vol 95 pp. 213-26
- van der Linden et al. Inter-organizational future proof EHR systems A review of the security and privacy related issues. Int J Med Inform (2009) Vol 78 (3) pp.141-60
- van der Linden et al. PropeR revisited. Int J Med Inform (2005) vol. 74 (2-4) pp. 235-44
- Tange et al. Towards a PropeR combination of patient records and protocols. Int J Med Inform (2003) vol. 70 (2-3) pp. 141-8
- van der Linden et al. PropeR: a multi disciplinary EPR system. Int J Med Inform (2003) vol. 70 (2-3) pp. 149-60

Abstracts

- van der Linden et al. Generic screen representations for future proof systems--is it possible? Two-model approach to a generic GUI. Stud Health Technol Inform (2007) vol. 129 (Pt 2) pp. 1122-6
- van der Linden et al. Towards a Generic Connection of EHR and DSS. Stud Health Technol Inform (2005) vol. 116 pp. 211-6
- van der Linden et al. Archetypes: the PropeR way. Medinfo (2004) vol. 11 (Pt 2) pp. 1110-4
- van der Linden et al. PropeR and archetypes. Stud Health Technol Inform (2004) vol. 110 pp. 49-53
- van der Linden et al. PropeR revisited. Stud Health Technol Inform (2003) vol. 95 pp. 346-51
- Boers et al. A distributed architecture for medical research. Stud Health Technol Inform (2002) vol. 90 pp. 734-8
- Tange et al. Towards a PropeR combination of patient records and protocols. Stud Health Technol Inform (2002) vol. 90 pp. 262-7
- van der Linden et al. An architecture for a virtual electronic health record. Stud Health Technol Inform (2002) vol. 90 pp. 220-5

Dankwoord

Toen ik in een onbewaakt ogenblik riep dat ik behalve de software ontwikkeling ook wel het onderzoek van het PropeR project wilde doen, waren Jan en Arie nogal sceptisch. Heren, bedankt dat jullie in al die jaren me toch geholpen hebben om dit eindpunt te bereiken. Bedankt ook voor alle eindeloze discussies. Gevreesd, verwarrend, grappig, maar uiteindelijk allemaal nuttig.

*A special word of thanks goes to Dipak Kalra. I'll never forget how you reacted after I told you my story at MIE*2006*. You stepped in and helped clarify things for me. And you stayed in the background, available whenever I called for your help. It's time to let everyone know that without your nudges, reflections and sheer positive attitude, this thesis would never have seen daylight.*

Verder wil ik mijn team van testers bedanken. Christel, Christien, Suzanne en Ine, heel hartelijk dank voor al jullie tijd en moeite en medewerking.

Patrick, jarenlang kamergenoot en vriend. Je luisterde als ik weer eens mijn ei kwijt moest en deed gezellig mee als ik op alles en iedereen afgaf. Je was stil als ik dat vroeg, maar kon me ook ongelofelijk aan het lachen krijgen met flauwe websites, grappige filmpjes en mopjes voor computer nerds. Dank je wel voor al die jaren, ik mis ze wel.

Bedankt ook al mijn collega's van Medische Informatica, Sjef, Huibert, Siebren, Ed en Marianne voor de fijne jaren in de vakgroep.

Agnieszka, we hadden vrij weinig met elkaar, tot die gedenkwaardige nacht tijdens de promovendi-dagen. Je werd een hele goede vriendin in die laatste jaren waarin we elkaar opjutten, troostten en motiveerden om dat eindpunt te bereiken. Ik ben er nu, hierna is het jouw beurt. Ik zal er voor zorgen dat jij er ook komt.

Vera, jouw verhalen over je zoon waren steeds weer een duidelijk voorbeeld waarom virtuele EPDs zo belangrijk zijn. Verder wil ik alle mensen bedanken die in de afgelopen jaren steeds belangstellend vroegen hoe het er mee stond. Nu kan ik eindelijk antwoorden dat het af is.

Pap en mam, bedankt voor de alle hulp. Jullie staan altijd klaar als we jullie nodig hebben. Alice, bedankt voor je geweldige inzet om van deze promotie een gedenkwaardige gebeurtenis te maken.

De laatste, maar zeker niet de onbelangrijkste, woorden zijn voor Paul, mijn rots in de branding. Schat, je belangstellende vragen kregen vaak een bits antwoord, maar ik was er toch blij mee. Net als met alle rustige acties als ik weer eens in de stress schoot. En dan zijn daar nog mijn kleine rotsjes, Bob en Luc. Jongens, jullie hebben lang moeten wachten, maar nu is de stress voorbij. Mama heeft het af.